Titre: Title:	Least squares design of recursive digital filters
Auteur: Author:	Xue Feng Wang
Date:	2003
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Wang, X. F. (2003). Least squares design of recursive digital filters [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7158/

UNIVERSITÉ DE MONTRÉAL

LEAST SQUARES DESIGN OF RECURSIVE DIGITAL FILTERS

WANG XUE FENG
DÉPARTEMENT DE GÉNIE ÉLECTRIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉLECTRIQUE)
FÉVRIER 2003

Your file Votre référence
ISBN: 0-612-86436-7
Our file Notre référence
ISBN: 0-612-86436-7

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

LEAST SQUARES DESIGN OF RECURSIVE DIGITAL FILTERS

Présenté par: WANG Xue Feng

en vue de l'obtention du diplôme de: Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de:

M. LEMIRE Michel, M.Ing., président

M. CORINTHIOS Michael J, Ph.D., membre et directeur de recherche

M. CONAN Jean, Ph.D., membre

Dedicated to my wife, my parents.

ACKNOWLEDGEMENT

The author would like to express his appreciation to his supervisor Professor Michael J. CORINTHIOS for the encouragement and guidance in the graduate studies in the Department of Electrical Engineering, École Polytechnique de Montréal, also thanks to Professor Michael J. CORINTHIOS for his valuable suggestion in the research of this thesis, carefully reading this thesis and correcting it.

Professor Michael J. CORINTHIOS has very profound academic attainments in spectrum analysis and mathematics model. The author has learned a lot from his supervision in the study of this project. It is also the author's pleasure to study and work under Professor Michael J. CORINTHIOS's supervision. All these will produce deep influence on the future study and work of the author.

Finally, the author would like to thank the jury, department staff, other peoples and friends in Electrical Engineering Department and École Polytechnique.

RÉSUMÉ

Au début de la thèse, des méthodes pour concevoir les filtres à réponse impulsionnelle infinie (RII) dans le domaine fréquentiel et le domaine temporel sont decrites.. Ensuite, l'auteur propose quatre algorithmes pour concevoir les filtres RII.

Parmi les algorithmes proposés, 3 (*l'algorithme 1*, *l'algorithme 2*, *l'algorithme 3*) sont basés sur la méthode de Gauss-Newton atténuée dans le domaine fréquentiel. Le quatrième algorithme (*l'algorithme 4*, c.-à-d., l'algorithme de QR-SVD PRONY) est basé sur les techniques d'analyse numérique et la méthode de Prony dans le domaine temporel. Tous ces algorithmes proposés emploient le critère des moindres carrés pour concevoir les filtres RII.

Dans le domaine fréquentiel, *l'algorithme 1* approxime la réponse arbitraire en amplitude du filtre désiré. *L'algorithme 2* et *l'algorithme 3* approximent simultanément la réponse arbitraire en amplitude et la réponse arbitraire en phase, avec la région stable indiquée. Les rayons maximums des pôles peuvent être fixés arbitrairement. Plusieurs exemples sont examinés pour comparer les algorithmes proposés avec d'autres algorithmes. Les résultats de comparaison montrent que *l'algorithme 1* a de bons résultats pour approximer la réponse arbitraire en amplitude. *L'algorithme 2* a de très bons résultats pour approximer la réponse arbitraire en amplitude et la réponse arbitraire en phase. *L'algorithme 3* a également de bons résultats, mais dans certains cas, *l'algorithme 3* n'a pas la bonne convergence. *L'algorithme 1* et *l'algorithme 2* ont non seulement de bons résultats mais ont également une bonne convergence.

Dans le domaine temporel, *l'algorithme 4* (l'algorithme de QR-SVD PRONY) approxime la réponse arbitraire impulsionnelle du filtre désiré, basé sur les techniques d'analyse numérique et la méthode de Prony. Dans cet algorithme, la factorisation de QR

et la décomposition de SVD sont employées pour calculer les coefficients du dénominateur et les coefficients du numérateur. Des exemples de conception sont utilisés pour comparer *l'algorithme 4* avec la fonction *prony*() de *Matlab* [Natic Mass, (1994)]. Cet algorithme a également de bons résultats et une bonne convergence.

ABSTRACT

At the beginning of this thesis, some methods to design IIR filters in the frequency domain and time domain are reviewed. Then four algorithms are proposed by the author to design IIR filters. Three of them (*Proposed algorithm 1, Proposed algorithm 2, Proposed algorithm 3*) are proposed in the frequency domain based on the damped Gauss-Newton method. The fourth (*Proposed algorithm 4*, i.e., *QR-SVD PRONY* algorithm) is proposed in the time domain based on numerical analysis techniques and Prony method. All these proposed algorithms use least squares error criterion to design IIR filter.

In the frequency domain, *Proposed algorithm 1* approximates the arbitrary magnitude response of the desired filter. *Proposed algorithm 2* and *Proposed algorithm 3* approximate simultaneously the arbitrary magnitude and phase response with the specified stable region, the maximum pole radius can be set arbitrarily. Several examples serve to compare the proposed algorithms with other algorithms. The comparison results show that *Proposed algorithm 1* has very good results for approximating arbitrary magnitude response. *Proposed algorithm 2* is shown to produce very good results for approximating arbitrary magnitude and phase response, and *Proposed algorithm 3* also has good results, but in some cases *Proposed algorithm 3* does not have good convergence. *Proposed algorithm 1* and *Proposed algorithm 2* not only have good results but also have good convergence.

In the time domain, *Proposed algorithm 4* (*QR-SVD PRONY* algorithm) approximates arbitrary time impulse response of the desired filter based on numerical analysis techniques and Prony method. In *Proposed algorithm 4*, QR factorization and SVD decomposition are used to compute the denominator coefficients and numerator coefficients. Design examples are used to compare *Proposed algorithm 4* with *Matlab prony*( ) function. *Proposed algorithm 4* also has good results and good convergence.

CONDENSÉ EN FRANÇAIS

Les signaux existent dans presque tous les domaines de la science, de la technologie et de nos vies quotidiennes. Des signaux peuvent être généralement divisés en deux classes: signaux à temps continus et signaux à temps discrets. Un signal à temps continu est défini à chaque instant du temps. Une forme d'onde de tension, une forme d'onde courante, et la vague saine sont des exemples typiques. D'autre part, un signal à temps discret est défini à un instant donné. Par exemple, la température haute\basse quotidienne, les données des disques audionumériques [ANDREAS ANTONIOU, (1993)].

Le filtrage est un processus par lequel le spectre fréquentiel d'un signal peut être traité, suivant les caractéristiques désirées [ANDREAS ANTONIOU, (1993)]. Les filtres numériques sont des systèmes numériques qui peuvent être employés pour filtrer des signaux à temps discrets. Les filtres numériques, au début, étaient des simulations des filtres analogiques sur les ordinateurs. Dans les années 1960 et 1970, quand la technologie de VLSI n'était pas encore bien développée, les filtres numériques étaient volumineux, coûtaient très cher, et consommaient trop de puissance. Dans les années 1980 et 1990, la technologie de VLSI s'est développée très rapidement. Maintenant, des filtres numériques peuvent être intégrés dans une carte très petite. La consommation d'énergie et le coût des filtres numériques sont nettement réduits [T.W. Parks, C.S. Burrus, (1987)].

Des filtres numériques sont conçus dans deux domaines: le domaine fréquentiel et le domaine temporel. Dans le domaine fréquentiel, étant donné la réponse en fréquence désirée du filtre (filtre numérique ou filtre analogique), le filtre numérique réel peut être conçu en se basant sur des méthodes des transformations traditionnelles (c.-à-d., les transformations du filtre analogue au filtre discret), ou sur quelques méthodes algorithmiques. Dans le domaine temporel, suivant la réponse arbitraire impulsionnelle

du filtre désiré, le filtre numérique réel peut être conçu en se basant sur quelques méthodes d'approximation ou méthodes de préfiltrage. D'une façon générale, les méthodes dans le domaine temporel peuvent seulement garantir que les filtres conçus ont la réponse en amplitude stable, mais elles ne garantissent pas la stabilité de la réponse en phase. Cependant, quelques méthodes de domaine fréquentiel peuvent garantir que les filtres conçus sont stables non seulement dans la réponse en amplitude, mais également dans la réponse en phase. Jusqu'ici, beaucoup de livres et articles sur le sujet des filtres à réponse impulsionnelle infinie (RII)ont été publiées.

Traditionnellement, dans le domaine fréquentiel, un filtre RII est conçu par la transformation d'un filtre analogique en filtre numérique. Les transformations classiques incluent la transformation d'invariance à l'impulsion, la transformation bilinéaire et les transformations spectrales. Les techniques classiques utilisent des approximations polynomial pour apparier les formes standard de filtre telles que passe-bas, passe-bande, passe-haut [Leland B. Jackson, (1989), ANDREAS ANTONIOU, (1993), Alan V. Oppenheim et Ronald W. Schafer, (1989), T.W. Parks, C.S. Burrus, (1987), etc.].

Ces approches sont utiles si le filtre conçu est un filtre du type sélectif en fréquence (Butterworth, Chebychev I et II, et filtre Elliptique), et qu'une solution existe. Cependant, les techniques classiques des transformations ont quelques inconvénients. Le premier inconvénient est que le filtre avec la forme arbitraire ou non classique (amplitude) ne peut pas être obtenu. Le deuxième inconvénient est que seulement la forme (amplitude) du filtre désiré peut être approximé , alors que la phase du filtre désiré ne peut pas être approximé.

Ces dernières années, beaucoup d'algorithmes parallèles et processeurs parallèles ont été proposés pour l'analyse général du spectre et identification des systèmes [Michael J. Corinthios, (1977), (1985), (1986), (1988), (1991), (1994)]. En 1996,

Michaël J. Corinthios a proposé un nouvel algorithme parallèle rapide de spectre en z pondéré et une architecture de processeur 1 D et 2 D pour la synthèse du filtre RII et l'identification des systèmes [Michael J. Corinthios, (1996)]. Cet algorithme [Michael J. Corinthios, (1996)] est efficace et a de très bonne performance dans la synthèse de filtre numérique et l'identification des systèmes.

Pour la plupart des cas, si un filtre de réponse en fréquence arbitraire (amplitude et phase) est conçu, une méthode algorithmique numérique est la seule approche possible. Une grande variété de méthodes algorithmiques numériques pour la synthèse de filtre RII ont été publiée. Selon un critère spécifique de norme d'erreur, ces méthodes algorithmiques peuvent être divisées en norme L_2 (technique d'approximation des moindres carrés), et norme L_∞ (technique d'approximation de norme de Chebychev). Pour concevoir un filtre de la norme L_2 ou un filtre de la norme L_∞, suivant la réponse en fréquence complexe, le critère suivant devra être résolu.

$$\min\left\|\varepsilon(\omega)\right\|_p = \min\left\|H(\omega) - H_d(\omega)\right\|_p = \min\left\|\frac{B(\omega)}{A(\omega)} - H_d(\omega)\right\|_p \qquad (1)$$

où $H(\omega)$ est la réponse en fréquence réelle de filtre, $H_d(\omega)$ est la réponse en fréquence désirée du filtre, $\left\|\varepsilon(\omega)\right\|_p$ est la norme p de l'erreur, $p = 2$, ou $p = \infty$.

Par exemple, x est un vecteur de dimension L, $x \in R^L$, la norme p de x est définie par:

$$\left\|x\right\|_p = \left(\sum_{i=1}^{L} |x_i|^p\right)^{\frac{1}{p}}, \qquad p \geq 1.$$

D'une façon générale, deux classes de norme p sont étudiés pour concevoir un filtre numérique, $p=2$ (norme L_2), et $p=\infty$ (norme L_∞ ou norme de Chebychev).

Pour $p=2$ $\quad \|x\|_2 = (\sum_{i=1}^{L}|x_i|^2)^{\frac{1}{2}}$

Pour $p=\infty$ $\quad \|x\|_\infty = \max|x_i| \quad (i=1,2,3,...,L)$

Selon la définition de la norme L_p, quand $p=2$, l'équation (1) s'appelle le problème des moindres carrés. Quand $p=\infty$, l'équation (1) s'appelle le problème d'approximation de norme de Chebychev. Ainsi, le problème (des moindres carrés) d'approximation de la norme L_2 est

$$\min\|\varepsilon(\omega)\|_2 = \min\|H(\omega)-H_d(\omega)\|_2 = \min\left\|\frac{B(\omega)}{A(\omega)}-H_d(\omega)\right\|_2$$

$$= \min\left\{\sum_{i=1}^{L}\left|\frac{B(\omega_i)}{A(\omega_i)}-H_d(\omega_i)\right|^2\right\}^{\frac{1}{2}} \qquad (2)$$

Le problème d'approximation de la norme L_∞ (la norme de Chebyshev) est

$$\min\|\varepsilon(\omega)\|_\infty = \min\|H(\omega)-H_d(\omega)\|_\infty = \min\left\|\frac{B(\omega)}{A(\omega)}-H_d(\omega)\right\|_\infty$$

$$= \min\ \max\left|\frac{B(\omega_i)}{A(\omega_i)}-H_d(\omega_i)\right| \qquad i=1,2,3,.....,L \qquad (3)$$

Au lieu de résoudre l'équation (1), Le critère suivant devrait être résoudre:

$$\min\|W(\omega)\varepsilon(\omega)\|_p = \min\|W(\omega)(H(\omega)-H_d(\omega))\|_p \qquad (4)$$

où $W(\omega)$ est la fonction de pondération. Quand $p=2$, l'équation (4) s'appelle le problème des moindres carrés pondéré.

Quelques articles sur le sujet des techniques des moindres carrés pondérés pour concevoir les filtres RII ont été publiés, comme [M. C. Lang, (2000), M. C. Lang, (1998)]. En raison de les non linéarité des équations (2) et (4), des méthodes d'optimisation, telles que des méthodes du type de Newton, doivent être employées.

Lang [Lang, (2000)] a employé la méthode de Gauss-Newton et un algorithme d'échange multiple pour résoudre l'équation (4) pour concevoir les filtres RII. Ces filtres ont des réponses en amplitude prescrites et des réponses en phase prescrite, et une contrainte de rayon de poteau. D'ailleurs, une région prescrite de stabilité peut être garantie, c.-à-d., le plus grand rayon de poteau peut être décidé à l'avance. Cependant, cette méthode a quelques inconvénients: le premier inconvénient est le choix de l'ordre du filtre, l'ordre du filtre conçu ne peut pas être choisi arbitrairement. Si l'ordre de filtre n'est pas choisi correctement, l'erreur du filtre conçu pourrait être grande. L'autre inconvénient est que cette méthode est compliquée mathématiquement.

Au lieu de résoudre l'équation non linéaire de problème (4), on pourrait linéarité l'équation (4) comme suit:

$$\min\left\|\varepsilon'(\omega)\right\|_2 = \min\left\|W(\omega)(B(\omega) - H_d(\omega)A(\omega))\right\|_2 \tag{5}$$

C'est un problème d'erreur d'équation linéaire pondéré. Quelques articles ont été publiés pour concevoir les filtres RII en résolvante de l'équation (5) [Wu-Sheng Lu, Soo-Chang Pei et Chien-Cheng Tseng, (1998), Wu-Sheng Lu, (1997), Y.C. Lim, J. H. Lee, C. K. Chen et R. H. Yang, (1992), kobayashi T. et S. Imai, (1990)].

Lu [Wu-Sheng Lu (1998)] a conçu les filtres (RII) stables de 1-D et 2-D en résolvante de l'équation (5) avec la technique de programmation quadratique.

Quelques articles ont été publiés pour concevoir le filtre RII en employant la technique d'approximation de Chebyshev, comme [XIANGKUN CHEN et THOMAS W. PARKS, (1990), A. G. Deczky, (1972), S. Alliney et F. Sgallari, (1980), Y. Ishizaki et H. Watanabe, (1968)]. Chen a conçu les filtres RII en utilisant le critère d'approximation d'erreur de Chebyshev en 1990 [Chen et PARKS, (1990)]. L'algorithme de Chen [Chen et PARKS, (1990)] a été basé sur la méthode d'Ellacott-Williams [S. Ellacott et J. Williams, (1976), J. Williams, (1979)]. L'algorithme de Chen conçoit séparément le numérateur et le dénominateur, et emploie une méthode de programmation linéaire pour résoudre le problème d'approximation linéaire complexe. La meilleure approximation locale a été trouvée habituellement dans cinq itérations et a eu une phase presque linéaire dans les bandes passantes. [Chen et PARKS, (1990)].

Le choix entre le critère de la norme L_2 et le critère d'erreur de la norme L_∞ est déterminé par l'application. Dans beaucoup de cas, le critère d'erreur de la norme L_2 est utile dans les bandes d'arrêt des filtres du type sélectifs en fréquence. Réduire au minimum l'erreur carrée moyenne dans les bandes d'arrêt réduit au minimum le gain de bruit dans ces bandes. Si un algorithme approprié est employé, la puissance de bruit peut être réduite au minimum, pourvu que son spectre de densité de puissance soit connu [Lang, (2000)]. Beaucoup d'articles ont employé le critère des moindres carrés pondéré (WLS) pour concevoir les filtres RII [Lang, (2000), Lu, (1998)]. Mais d'après sur la recherche que nous avons fait, nous avons constaté que le critère WLS n'est pas un bon choix pour concevoir les filtres RII. Au lieu de WLS, le critère des moindres carrés (LS) est choisi dans cette thèse, c.-à-d., réduire au minimum l'équation (2) pour concevoir les filtres RII. Les raisons pour laquelle le critère LS est choisi sont les suivantes :

- Il est difficile de trouver les fonctions de pondération appropriées $W(\omega)$ pour satisfaire toutes les caractéristiques de bandes d'un filtre désiré. D'ailleurs, pour des filtres différents, nouvelles fonctions de pondération devraient être choisies.

- Plusieurs exemples montrent que l'utilisation du critère WLS ne donne pas de meilleurs résultats que ceux du critère LS. Dans certains cas, employer le critère WLS peut donner de plus mauvais résultats.

Pour les raisons mentionnées ci-dessus, Le critère de LS est choisi pour concevoir les filtres RII dans cette thèse.

Dans le domaine temporel, il y a principalement deux méthodes pour concevoir le filtre RII. La première est la méthode Prony [T.W. Parks et C.S. Burrus, (1987), C. S. Burrus et T.W. Parks, (1970)], l'autre est la méthode de Steiglitz-Mcbride [L.E. McBride, H.W. SCHAEFGEN et K. STEIGLITZ, (1966)]. La plupart des formulations de conception de domaine temporel des filtres RII donnent des équations non linéaires.

Prony, en 1790, a trouvé une formulation spéciale pour analyser les propriétés élastiques des gaz, qui ont produit des équations linéaires. Une forme plus générale que la méthode de Prony peut être appliquée à la conception de filtre RII en réduisant au minimum l'erreur de solution. La réponse impulsionnelle temporel du filtre désiré est donnée, puis en résolvante de l'équation linéaire, les coefficients du dénominateur et les coefficients du numérateur peuvent être obtenus. La méthode de Prony est une bonne méthode pour concevoir les filtres RII dans le domaine temporel, et c'est également une méthode de conception efficace [T. W. Parks et C.S. Burrus, (1987)].

La méthode de Steiglitz-Mcbride est une méthode de préfiltrage itérative. Un ensemble d'équations linéaires pour les coefficients de dénominateur et les coefficients de numérateur qui réduisent au minimum l'erreur de l'équation peut être résolue. Les coefficients du filtre conçu peuvent être obtenus [L.E. McBride, H.W. SCHAEFGEN et K. STEIGLITZ, (1966), MONSON H. HAYES, (1996)].

Dans cette thèse, quatre algorithmes pour concevoir les filtres RII sont proposés. Trois (*l'algorithme 1*, *l'algorithme 2*, et *l'algorithme 3*) sont proposés dans le domaine de fréquence pour concevoir les filtres RII basés sur la méthode de Gauss-Newton atténuée. *L'algorithme 3* utilise la méthode de Gauss-Newton atténué et la programmation quadratique.

Le quatrième algorithme est proposé (*l'algorithme 4*, c.-à-d., l'algorithme de QR-SVD-PRONY) dans le domaine temporel pour concevoir les filtres RII basés sur des techniques d'analyse numérique et la méthode de Prony.

L'algorithme 1 conçoit les filtres stables RII en rapprochant la réponse arbitraire en amplitude des filtres désirés. Quelques exemples sont employés pour comparer *l'algorithme 1* avec d'autres algorithmes. Des résultats de comparaison sont donnés sous forme de tables et de figures. Les résultats de comparaison montrent que *l'algorithme 1* a des très bons résultats et une bonne convergence.

L'algorithme 2 et *l'algorithme 3* rapprochent simultanément la réponse arbitraire en amplitude et la réponse arbitraire en phase du filtre désiré avec la région de stabilité prescrite. Quelques exemples sont employés pour comparer *l'algorithme 2*, et *l'algorithme 3* avec d'autres algorithmes. Des résultats de comparaison sont donnés. Les résultats de comparaison montrent que *l'algorithme 2* a des très bons résultats et une bonne convergence, *l'algorithme 3* a également de bons résultats, mais dans certains cas, il n'a pas la bonne convergence. Le choix de l'ordre du filtre est facile et sans contrainte.

D'ailleurs, les rayons maximums des pôles être fixé arbitrairement à l'avance, c.-à-d., le filtre avec la région de stabilité prescrite peut être obtenu.

L'algorithme 4 (l'algorithme de QR-SVD PRONY) conçoit les filtres RII stable dans le domaine de temps basé sur quelques techniques d'analyse numérique (factorisation de QR et décomposition de SVD) et la méthode de Prony. Des exemples sont également employés pour comparer *l'algorithme 4* avec la fonction *prony*( ) de *Matlab* [Natic Mass, 1994]. Des résultats de comparaison sont donnés. Les résultats de comparaison montrent que *l'algorithme 4* a des très bons résultats et une très bonne convergence.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

$A(\omega)$	filter denominator polynomial
$B(\omega)$	filter numerator polynomial
$e(n)$	filter time impulse response error
E_1	filter magnitude error
$E(x)$	filter frequency response error
g	Gradient
$h(n)$	the actual impulse response of the filter
$h_d(n)$	the desired impulse response of the filter
H	Hessian matrix
$H(\omega)$	actual filter frequency response
$H_d(\omega)$	desired filter frequency response
J	Jacobi matrix
QR	QR factorization
$\mathrm{Re}(x)$	the real component of x
SVD	singular value decomposition
$W(\omega)$	weighting function
$\nabla^2 E(x)$	Hessian matrix

INTRODUCTION

Signals exist in almost every field of engineering, science and our daily lives, such as in communications, measurement science, automatic control engineering, medical science, biomedical engineering, etc. Signals can be generally divided into two classes, i.e., continuous-time signals and discrete-time signals. A continuous-time signal is one that is defined at each instant of time. For example, a voltage waveform, a current waveform, and noise wave are continuous-time signal. A discrete-time signal is one that is defined at discrete instants of time, for example, the computer disc data, the output of A/D converter [ANDREAS ANTONIOU, (1993)].

Filtering is a process by which the frequency spectrum of a signal can be processed according to some desired specifications. Digital filters are digital systems that can be used to filter discrete-time signals. In the 1960's and 1970's, when VLSI technology was not developed well, digital filters were very large, very inconvenient to use. In the 1980's and 1990's, VLSI technology ands DSP technology developed very fast. Now digital filters can be made into a very small chip. The cost and power consumption of digital filters are largely reduced. This has led to widespread application of digital filters [T. W. Parks and C. S. Burrus, (1987)].

Digital filters are designed in two domains: the frequency domain and the time domain. In frequency domain, given the desired filter (digital filter or analog filter) frequency response, the actual digital filter can be designed based on some traditional transform methods (i.e., the transforming from analog filter to discrete filter), or based on some algorithmic methods. In time domain, given the desired filter time impulse response, the actual digital filter can be designed based on some approximation methods or prefiltering methods. Using which domain design methods depends on the design problems. Generally, time domain design methods can only guarantee that the filters to

be designed have stable magnitude response, but they do not guarantee stable phase response. However some frequency domain design methods can guarantee the filters to be designed be stable not only in the magnitude response but also in the phase response. So far, many books and papers about the designs of digital IIR filters have been published.

Traditionally, in frequency domain, IIR filters are designed by transforming an analog filter into a digital filter. The classical transformations involve Impulse Invariance transformation and bilinear transformation and spectral transformations. Classical designs utilize polynomial approximations to match standard filter shapes such as low-pass, band-pass, high-pass, etc, [Leland B. Jackson, (1989), ANDREAS ANTONIOU, (1993), Alan V. Oppenheim and Ronald W. Schafer, (1989), T. W Parks and C. S. Burrus, (1987), etc.].

These design approaches are useful if the designed filter is a frequency-selective filter (the Butterworth, the Chebyshev I and II, and Elliptic filter), and the solutions exist. However, classical transformation techniques have some drawbacks. The first drawback is that the filter with arbitrary or non-classical shape (magnitude) cannot be obtained. The second drawback is that only the shape (magnitude) of the desired filter can be matched, while the phase of the desired filter cannot be matched.

In recent years, many parallel algorithms and parallel processors for the generalized spectrum analysis and system identification have been proposed. The z-transform plays a very important role in these applications. Z-transform contour evaluation has also been applied to estimate the pole-zero of the digital system by using the radial and circular "discrete z-transforms" (DZTS) [Michael J. Corinthios, (1977), (1985), (1986), (1988), (1991), (1994)].

In 1996, Michael J. Corinthios proposed a new fast weighted z spectrum parallel algorithm and 1 D and 2 D parallel processor architectures for the IIR filter synthesis and system identification [Michael J. Corinthios, (1996)]. The proposed weighted z transform spectrum can evaluate the pole-zero mathematical model of an IIR system. This algorithm [Michael J. Corinthios, (1996)] is efficient and has very good performance in digital filter design and system identification. It is different from digital filter synthesis techniques [T. W. Parks, (1987), L. R. Rabiner, J. H. McClellan, and T.W. Parks, (1975)].

Besides the parallel weighted z spectrum technique [Michael J. Corinthios, (1996)] and the classical transformation techniques[Leland B. Jackson, (1989), ANDREAS ANTONIOU, (1993), Alan V. Oppenheim and Ronald W. Schafer, (1989), Parks T. W, and C. S. Burrus, (1987), etc.], a numerical algorithm approach is also used to design IIR filter.

A wide variety of numerical algorithmic methods for IIR filters designs have been reported [Mathias C. Lang, (2000), X. Chen and T. W. Parks, (1990), etc.]. According to a specific error norm criterion, these algorithmic methods can be divided into L_2 norm (Least squares approximation technique), and L_∞ norm (Chebyshev norm approximation technique). To design an L_2 norm or an L_∞ norm filter, given a complex frequency response, the following criterion is sought:

$$\min\left\|\varepsilon(\omega)\right\|_p = \min\left\|H(\omega) - H_d(\omega)\right\|_p = \min\left\|\frac{B(\omega)}{A(\omega)} - H_d(\omega)\right\|_p \qquad (1)$$

where $H(\omega)$ is the actual filter frequency response, $H_d(\omega)$ is the desired filter frequency response, $B(\omega)=\sum_{m=0}^{M} b_m e^{-j\omega m}$, $A(\omega)=1+\sum_{n=1}^{N} a_n e^{-j\omega n}$, $\|\varepsilon(\omega)\|_p$ is the p norm of the error $\varepsilon(\omega)$, $p=2$, or $p=\infty$.

For example, x is a L-dimension vector, $x\in R^L$, the p norm of x is defined as:

$$\|x\|_p=(\sum_{i=1}^{L}|x_i|^p)^{\frac{1}{p}}, \qquad p\geq 1.$$

Generally, two kinds of p norm are studied to design a digital filter. That is $p=2$ (L_2 norm), and $p=\infty$ (L_∞ norm or Chebyshev norm).

For $p=2$ $\|x\|_2=(\sum_{i=1}^{L}|x_i|^2)^{\frac{1}{2}}$

For $p=\infty$ $\|x\|_\infty=\max|x_i|$ $(i=1,2,3,...,L)$

When $p=2$, according to the definition of L_p norm, equation (1) is usually called least squares problem. When $p=\infty$, equation (1) is usually called Chebyshev norm approximation problem. Thus, L_2 norm (Least Squares) approximation problem is

$$\min\|\varepsilon(\omega)\|_2=\min\|H(\omega)-H_d(\omega)\|_2=\min\left\|\frac{B(\omega)}{A(\omega)}-H_d(\omega)\right\|_2$$

$$=\min\left\{\sum_{i=1}^{L}\left|\frac{B(\omega_i)}{A(\omega_i)}-H_d(\omega_i)\right|^2\right\}^{\frac{1}{2}} \qquad (2)$$

The L_∞ norm(Chebyshev norm) approximation problem is

$$\min\left\|\varepsilon(\omega)\right\|_\infty = \min\left\|H(\omega)-H_d(\omega)\right\|_\infty = \min\left\|\frac{B(\omega)}{A(\omega)}-H_d(\omega)\right\|_\infty$$

$$= \min \max\left|\frac{B(\omega_i)}{A(\omega_i)}-H_d(\omega_i)\right| \qquad \text{for} \quad i=1,2,3,\ldots\ldots,L \tag{3}$$

Instead of solving equation (1), the following problem should be solved:

$$\min\left\|W(\omega)\varepsilon(\omega)\right\|_p = \min\left\|W(\omega)(H(\omega)-H_d(\omega))\right\|_p \tag{4}$$

where $W(\omega)$ is the weighting function. When $p=2$, equation (4) is called the weighted least squares problem.

Some papers using weighted least squares techniques to design IIR filters have been published, such as [M.C. Lang, (1998), (2000)]. Because of the non linearity of equation (2), (4), optimization methods such as Newton type methods have to be used. Lang [Lang, (2000)] used Gauss-Newton method and a multiple exchange algorithm to solve equation (4) to design IIR filters with prescribed magnitude and phase responses and a pole radius constraint. This method can approximate prescribed magnitude and phase responses. Moreover, a prescribed stability region can be guaranteed, i.e., the largest pole radius can be decided in advance. However, this method also has some drawbacks: one is the choice of the filter order. The designed filter order cannot be chosen arbitrarily. If the filter order is not chosen properly, the error of the designed filter could be large. The other drawback is that this method is complicated in computation.

Instead of solving the nonlinear problem equation (4), one could linearize equation (4) as follows:

$$\min\left\|\varepsilon'(\omega)\right\|_2 = \min\left\|W(\omega)(B(\omega) - H_d(\omega)A(\omega))\right\|_2 \tag{5}$$

This is a weighted linear equation error problem. Some papers were published to design IIR filters by solving equation (5) [Wu-Sheng Lu, Soo-Chang Pei and Chien-Cheng Tseng, (1998), Wu-Sheng Lu, (1997), Y.C. Lim and J. H. Lee and C. K. Chen and R. H. Yang, (1992), T. kobayashi and S. Imai, (1990)]. Lu [Wu-Sheng Lu (1998)] designed stable 1-D and 2-D IIR filters by solving equation (5) with quadratic programming technique.

Some papers were published to design IIR filter by using Chebyshev approximaton technique, such as [XIANGKUN CHEN and THOMAS W. PARKS, (1990), A. G. Deczky, (1972), S. Alliney and F. Sgallari, (1980), Y. Ishizaki and H. Watanabe, (1968)]. Chen [Chen and PARKS, (1990)] designed IIR filter using Chebyshev approximation error criterion. The algorithm that Chen [Chen (1990)] proposed was based on the Ellacott-Williams method [S. Ellacott and J. Williams, (1976), J. Williams, (1979)]. The algorithm separately designs the numerator and denominator, and uses a linear programming method to solve the linear complex approximation problem. The locally best approximation was found usually in five iterations and had an exactly equiripple error and nearly linear phase in the passbands [Chen (1990)].

The choice between L_2 norm error criterion and L_∞ norm error criterion is determined by the application. In many cases, the L_2 norm error criterion (least squares or weighted least squares) is useful in the stopbands of frequency selective filters. Minimizing the L_2 error in the stopbands minimizes the noise gain in these bands. If a proper algorithm is used, the noise power can be minimized [Lang, (2000)]. In this thesis, the least squares criterion is chosen to design IIR filter.

In the time domain, there are mainly two methods to design IIR filter. One is prony method [Parks T.W. and C.S. Burrus, (1987), C. S. Burrus and Parks T.W., (1970)], the other is Steiglitz-Mcbride method [L.E. McBride, JR. and H.W. SCHAEFGEN AND K. STEIGLITZ, (1966)]. Most formulations of time-domain design of IIR filters give nonlinear equations.

Prony, in 1790, developed a method to analyze elastic properties of gases. A more general form of Prony's method can be applied to design IIR filter with a prescribed time domain finite impulse response. The time impulse response of the desired filter is given, then by solving linear equation, the denominator coefficients and numerator coefficients can be obtained respectively. The advantage of Prony method is that it develops linear method to resolve the nonlinear IIR filter design problem. Prony method is a successful method to design IIR filter in the time domain, and it is also an effective design method [T. W. PARKS AND C. S. BURRUS, (1987)].

Steiglitz-Mcbride method is a iterative prefiltering method. A set of linear equations for the denominator coefficients and numerator coefficients that minimize equation error can be derived by differentiating this error with respect to the coefficients. Differentiating and setting the derivatives equal to zero, then a set of linear equations can be obtained. These linear equations are a set of overdetermined linear equations. In the iterative prefilting method, by finding the least squares solution to the set of overdetermined linear equation, the equation error can be minimized. The coefficients of the designed filter can be obtained [L.E. McBride, JR. and H.W. SCHAEFGEN AND K. STEIGLITZ, (1966), MONSON H. HAYES, (1996)].

In this thesis, firstly, the frequency domain least squares IIR filter design methods are studied. Three algorithms(*Proposed algorithm 1, Proposed algorithm 2, Proposed algorithm 3*) are proposed to design IIR filters with arbitrary frequency response and

prescribed stability region based on damped Gauss-Newton method. *Proposed algorithm 3* combines damped Gauss-Newton method and quadratic programming.

Secondly, the time domain least squares IIR filters design method (Prony method) is studied. One algorithm (*Proposed algorithm 4*, i.e., *QR-SVD-PRONY*) method is proposed based on numerical analysis techniques and Prony method. Several design examples are tested to compare the proposed algorithms with other methods. The comparison results are given. The proposed algorithms are shown to obtain better results than other methods. All proposed algorithms have been implemented in *Matlab*[6.1].

CHAPTER I

LEAST SQUARES DESIGNS OF DIGITAL FILTERS IN THE FREQUENCY DOMAIN

Digital filters can be divided into two classes of filters, i.e., FIR (finite duration impulse-response) and IIR (infinite duration impulse-response) filters. The transfer function of digital filters is defined as the z transform of the unit-pulse response of the filter. Generally the digital filters can be assumed to be causal and can be expressed as a transfer function:

$$H(z)=\frac{b_0+b_1z^{-1}+b_2z^{-2}+.....+b_Mz^{-M}}{1+a_1z^{-1}+a_2z^{-2}+.....+a_Nz^{-N}} \tag{1.1}$$

The region of convergence of $H(z)$ lies outside a circle centred at the origin of the z plane. The poles of the filter lie inside or on this circle. For stable filters, the radius of this circle is less than unity, i.e., the poles of stable filters must lie inside the unit circle. If the transfer function of a filter can be written as a polynomial (all $a_i=0$), the filter has a finite-duration unit-pulse response and is called an FIR filter. If coefficients a_i are not equal to zero, then the filter has an infinite-duration unit-pulse response and is called an IIR filter. The design problem for a digital filter is the approximation problem of the desired filter. Given the desired filter, the coefficients a_i, b_i of the designed filter are obtained through optimal approximation of the desired filter. The approximation problem is usually stated in the frequency domain, equivalent to $z=e^{j\omega}$. The filter parameters to be chosen are the filter coefficients a_i, b_i in

$$H(e^{j\omega})=\frac{b_0+b_1e^{-j\omega}+b_2e^{-2j\omega}+.....+b_Me^{-jM\omega}}{1+a_1e^{-j\omega}+a_2e^{-2j\omega}+.....+a_Ne^{-jN\omega}}$$

In most cases, the approximation problem is a polynomial (for FIR filters) or rational (for IIR filters) approximation problem with a complex desired function on the frequency band from $-\pi$ to π. If the desired frequency response is symmetric about the frequency axis, the frequency band from 0 to π is often chosen [T. W. Parks, C. S. Burrus, (1987)]. The parameters a_i and b_i are chosen to minimize the difference between the desired response $H_d(z)$ and the actual response $H(z)$, often the p norm of the difference,

$$\|E(z)\|_p = \|H(z) - H_d(z)\|_p \tag{1.2}$$

where $\|\cdot\|_p$ is the p norm of a vector. For example x is a L-dimension vector $x \in R^L$, the p norm of x is defined as

$$\|x\|_p = (\sum_{i=1}^{L} |x_i|^p)^{\frac{1}{p}} \ , \ p \geq 1. \tag{1.3}$$

Generally, three kinds of p norm are studied i.e., $p=1$ (L_1 norm), $p=2$ (L_2 norm), $p=\infty$ (L_∞ norm i.e., chebyshev norm).

for $p=1$ $$\|x\|_1 = \sum_{i=1}^{L} |x_i| \tag{1.4}$$

for $p=2$ $$\|x\|_2 = (\sum_{i=1}^{L} |x_i|^2)^{\frac{1}{2}} \tag{1.5}$$

for $p=\infty$ $$\|x\|_\infty = \max|x_i| \qquad i = 1, 2, 3, ..., L \tag{1.6}$$

The L_2 norm is meaningful in many numerical applications, and easy to be computed. We often use L_2 norm to minimize the difference of equation (1.2), thus leading to the non-linear least squares problem:

$$\min\left\|E(z)\right\|_2 = \min\left\|\frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + \ldots\ldots + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots\ldots + a_N z^{-N}} - H_d(z)\right\|_2 \qquad (1.7)$$

The problem of designing a digital filter is to estimate the values of a_i, b_i through minimizing $\left\|E(z)\right\|_2$.

The traditional approach to the frequency domain design of IIR filters is the transforming an analog filter into a digital filter. The transformations involve Impulse Invariance transformation, bilinear transformation and spectral transformations. These design approaches are useful if the designed filter is the frequency-selective filter (the Butterworth, Chebyshev I and II, and Elliptic filter), and the solutions exist.

In most cases, if an arbitrary frequency response IIR filter is designed, a numerical algorithmic method is the only possible approach. That is using a numerical algorithmic method to estimate a_i and b_i in equation (1.2) through minimizing $E(z)$. Many numerical methods for the frequency domain design of IIR filters have been proposed. The intentions of developing these methods are mainly the followings [Lang , (2000)]:

a) Be interested in the magnitude specifications of the designed filter, not interested in the phase or group delay specifications.
b) Satisfy the magnitude specifications and phase specifications or group delay specifications simultaneously.
c) Use certain structures, for example cascade structures for implementation.

In this thesis, items a) and b) are applied to design IIR filter. In the frequency domain, two kinds of filters are designed:

1. Stable IIR filters approximating ideal magnitude response, not interested in phase response.
2. Stable IIR filters approximating simultaneously magnitude response and phase response with prescribed stability region.

Almost most papers which have been published used the weighted least squares criterion. Let $H_d(e^{j\omega})$ be the ideal frequency response, and $H(e^{j\omega})$ be the frequency response of the designed filter. According to the weighted least squares error criterion, that is:

$$\min \sum_{i=1}^{L} \left| W(e^{j\omega_i})[H(e^{j\omega_i}) - H_d(e^{j\omega_i})] \right|^2 \tag{1.8}$$

where $W(e^{j\omega_i})$ is the weighting function. But, in practice, we find weighted least squares error criterion is not a good choice for designing a IIR filter. The reasons are the followings: Firstly, it is difficult to find the proper weighted function corresponding to the different frequency band (pass band, transaction band, stop band). Secondly, several design examples show that using weighted least squares criterion can not obtain better results than simply using least squares criterion. In some cases, it even obtains worse results than using least squares criterion. Based on the reasons mentioned above, least squares error criterion is utilized to design IIR filters in this thesis.
That is

$$\min \sum_{i=1}^{L} \left| H(e^{j\omega_i}) - H_d(e^{j\omega_i}) \right|^2 \tag{1.9}$$

Because of the non-linearity of least squares error criterion, the optimization method has to be used. Usually the optimization methods are iterative, they often involve a large amount of computation. However, they are suitable for the design of filters which have arbitrary magnitude response or phase response. Moreover, they often

give high-quality designs. Initial guess which are close to the optimal solution for the iterative procedures are required. Here the linear equation error criterion is used to obtain the initial guess.

Let

$$E = \sum_{i=1}^{L} \left| H(e^{j\omega_i}) - H_d(e^{j\omega_i}) \right|^2 \tag{1.10}$$

where

$$H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})} = \frac{\sum_{m=0}^{M} b_m e^{-jm\omega}}{1 + \sum_{n=1}^{N} a_n e^{-jn\omega}} \tag{1.11}$$

a_n, b_m are unknown constants, which are to be estimated. ω is equally sampled at L frequency points ω_i between 0 and π. M and N are the order of the proposed filer. L is the number of the sampled frequency points.

In this thesis, the orders of the proposed filter M and N can be chosen arbitrarily, and need not to be equal. However, the values M and N of the filter cannot be chosen very large, or very small. If M and N are very large, then the computation of the filter coefficients a and b and the filter implementation become increasingly complex.. The values of M and N cannot be chosen very small either. For example, $M = N = 1$. If the values of M and N are very small, the proposed filter cannot work well. In this thesis, the values of M and N are chosen between 4 and 20. The values of the sampled frequency points L cannot be set very small. If the values of the sampled frequency points L are very small, then the sampled values of $H_d(\omega)$ cannot match the actual values of $H_d(\omega)$.We cannot obtain the correct values of $H_d(\omega)$. In this thesis, the values of the sampled frequency points L are greater than or equal to 10.

Define a vector:

$$x = [b_0, b_1, b_2, b_3, ..., b_M, a_1, a_2, a_3, ..., a_N]^T \tag{1.12}$$

where a, b are the coefficient variables in (1.11). We will use $H_i(x)$ to denote $H(e^{j\omega_i})$ and $E(x)$ to denote E in (1.10).

Hence,

$$E(x) = \sum_{i=1}^{L} \left| H_i(x) - H_d(e^{j\omega_i}) \right|^2 \tag{1.13}$$

or

$$E(x) = \sum_{i=1}^{L} f_i^{\,2}(x) \tag{1.14}$$

where

$$f_i(x) = \left| H_i(x) - H_d(e^{j\omega_i}) \right| \tag{1.15}$$

The filter coefficients vector x can be obtained through minimizing equation (1.13).

1.1 Least squares linear equation error IIR filters design

Ashraf Alkhairy [Ashraf Alkhairy,(1994)] and Tatsuya MATSUNAGA [Tatsuya MATSUNAGA,(2000)] proposed two algorithms based on the linearization of the nonlinear filter design problems. The following algorithm is based on the Ashraf algorithm [Ashraf Alkhairy,(1994)] and Tatsuya algorithm [Tatsuya MATSUNAGA,(2000)].

Let $$E'(e^{j\omega}) = H(e^{j\omega}) - H_d(e^{j\omega}) = \frac{B(e^{j\omega})}{1 + A'(e^{j\omega})} - H_d(e^{j\omega})$$

where $$A'(e^{j\omega}) = \sum_{n=1}^{N} a_n e^{-j\omega n}$$

then $$E'(e^{j\omega})[1 + A'(e^{j\omega})] = B(e^{j\omega}) - H_d(e^{j\omega})[1 + A'(e^{j\omega})]$$

$$= B(e^{j\omega}) - H_d(e^{j\omega}) - H_d(e^{j\omega})A'(e^{j\omega})$$

In order to minimize

$$\left\| B(e^{j\omega}) - H_d(e^{j\omega}) - H_d(e^{j\omega})A'(e^{j\omega}) \right\|_2$$

i.e., minimize

$$\left\| H_d(e^{j\omega}) + H_d(e^{j\omega})A'(e^{j\omega}) - B(e^{j\omega}) \right\|_2$$

let

$$H_d(e^{j\omega}) + H_d(e^{j\omega})A'(e^{j\omega}) - B(e^{j\omega}) = 0$$

then

$$H_d(e^{j\omega}) = B(e^{j\omega}) - H_d(e^{j\omega})A'(e^{j\omega})$$

where $$B(e^{j\omega}) = P_1 b$$

$$A'(e^{j\omega}) = P_2 a$$

and

$$a = [a_1, a_2, a_3, ..., a_N]^T$$

$$b = [b_0, b_1, b_2, b_3, ..., b_M]^T$$

$$P_1 = \begin{pmatrix} 1 & e^{-j\omega_1} & ..... & e^{-jM\omega_1} \\ 1 & e^{-j\omega_2} & ..... & e^{-jM\omega_2} \\ : & : & : & : \\ : & : & : & : \\ 1 & e^{-j\omega_L} & ..... & e^{-jM\omega_L} \end{pmatrix}$$

$$P_2 = \begin{bmatrix} e^{-j\omega_1} & e^{-j2\omega_1} & \cdots\cdots & e^{-jN\omega_1} \\ e^{-j\omega_2} & e^{-j2\omega_2} & \cdots\cdots & e^{-jN\omega_2} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ e^{-j\omega_L} & e^{-j2\omega_L} & \cdots\cdots & e^{-jN\omega_L} \end{bmatrix}$$

then

$$\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix} \begin{bmatrix} b \\ \\ a \end{bmatrix} = H_d(e^{j\omega}) \tag{1.16}$$

This is a least squares problem.

$\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}$ is a $L\times(M+N+1)$ matrix. $\begin{bmatrix} b \\ \\ a \end{bmatrix}$ is a $(M+N+1)\times 1$ vector.

$H_d(e^{j\omega})$ is a $L\times 1$ vector. Generally, $\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}$ is rectangular with $L > M+N+1$. Equation (1.16) can be solved in a least squares sense by solving the normal equations:

$$\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix} \begin{bmatrix} b \\ \\ a \end{bmatrix} = \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T H_d(e^{j\omega})$$

If $\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}$ is not singular, the solution is

$$\begin{bmatrix} b \\ \\ a \end{bmatrix} = \left\{ \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix} \right\}^{-1} \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T H_d(e^{j\omega})$$

It is not necessary to explicitly compute $\left\{ \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}^T \begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix} \right\}^{-1}$, Numerical analysis shows that it may lead bad results to explicitly compute the inverse of a matrix. Instead, QR factorization of $\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}$ or Cholesky factorization of $\begin{bmatrix} P_1 & -H_d(e^{j\omega})\,P_2 \end{bmatrix}$ can be used to solve equation (1.16). Also, Matlab matrix

operation "\" left division symbol can be used. The followings are the explanation of Matlab "\" left matrix division operator. According to [Natic Mass, (1994)], " $A\backslash B$ is the matrix left division of A into B, which is roughly the same as $A^{-1}*B$. If A is an N-by-N matrix, and B is a column vector with N components, or a matrix with several such columns, then $x = A \backslash B$ is the solution of the equation $A*x=B$ computed by Gaussian elimination. If A is an M-by-N matrix with $M <$ or $> N$ and B is a column vector with M components, or a matrix with several such columns, then $x = A \backslash B$ is the solution in the least squares sense to the underdetermined or overdetermined system of equation $A*x=B$" [Natic Mass, (1994)].

Then, from equation (1.16), the solution of vector $\begin{bmatrix} b \\ \\ a \end{bmatrix}$ is

$$\begin{bmatrix} b \\ \\ a \end{bmatrix} = \begin{bmatrix} P_1 & -H_d(e^{j\omega})\, P_2 \end{bmatrix} \backslash H_d(e^{j\omega})$$

Here a,b which are obtained are complex values. In order to obtain real coefficients, the real components of a and b are taken. This is the linear equation error IIR filter design. The advantages of this algorithm are:

- It is simple to obtain a,b , and the algorithm is efficient.
- The results which are obtained are very good. The linear equation error E' and the non-linear error E are very small.

The drawback of this algorithm is that the filter designed using this algorithm is not guarantied to be stable. The coefficients a,b which are obtained from this algorithm can be used as the initial guess of the optimization method.

1.2 Gauss-Newton method and its modifications

1.2.1 The Newton's method

The optimization problem of equation (1.14) can be solved by using the unconstrained optimization algorithms. In recent years, various classes of these algorithms have been proposed, such as steepest-descent algorithms, conjugate-direction algorithms and quasi-Newton algorithm [ANDREAS ANTONIOU, (1993)]. Among these algorithms, an important class of optimization algorithms: Gauss-Newton algorithm and its modifications are very effective for the design of IIR filters. Gauss-Newton type methods are general optimization methods. They are based on Newton's method for finding the minimum in quadric functions.

Considering equation (1.14), $E(x)$ should be minimized. If function $E(x)$ is a second-order differential function, the followings are necessary conditions for x being a local minimum of $E(x)$.

- Gradient $g = \nabla E(x) = 0$ at x, i.e.,

$$\frac{\partial E}{\partial x_i} = 0, \qquad (i = 1,2,3,.....,M+N+1) \tag{1.17}$$

- Hessian matrix $\nabla^2 E(x)$, i.e.,

$$H = \begin{pmatrix} \frac{\partial^2 E}{\partial x_1^{\,2}} & \frac{\partial^2 E}{\partial x_1 \partial x_2} & ..... & \frac{\partial^2 E}{\partial x_1 \partial x_{M+N+1}} \\ \frac{\partial^2 E}{\partial x_1 \partial x_2} & \frac{\partial^2 E}{\partial x_2^{\,2}} & ..... & \frac{\partial^2 E}{\partial x_2 \partial x_{M+N+1}} \\ \vdots & \vdots & ..... & \vdots \\ \vdots & \vdots & ..... & \vdots \\ \frac{\partial^2 E}{\partial x_1 \partial x_{M+N+1}} & \frac{\partial^2 E}{\partial x_2 \partial x_{M+N+1}} & ..... & \frac{\partial^2 E}{\partial x_{M+N+1}^{\,2}} \end{pmatrix} \tag{1.18}$$

is a positive semi-definite matrix at x i.e., $y^H Hy \geq 0$ for any vector $y \neq 0$, where y^H is the conjugate transpose of y.

On the other hand, the followings are the sufficient conditions for x being a local minimum of $E(x)$:

- Gradient $\nabla E(x) = 0$.
- Hessian $\nabla^2 E(x)$ is a positive definite matrix.

  i.e., $y^H Hy > 0$ for any vector $y \neq 0$, where y^H is the conjugate transpose of y.

Taylor series of $E(x)$ at $x + \Delta x$ is given by:

$$E(x+\Delta x) = E(x) + \sum_{i=1}^{M+N+1} \frac{\partial E(x)}{\partial x_i}\Delta x_i + \frac{1}{2}\sum_{i=1}^{M+N+1}\sum_{j=1}^{M+N+1} \frac{\partial^2 E(x)}{\partial x_i \partial x_j}\Delta x_i \Delta x_j + O(\|\Delta x\|_2^{\,2}) \quad (1.19)$$

where $O(\|\Delta x\|_2^{\,2})$ is the third and higher order terms. Since $E(x)$ is a quadratic function, its second partial derivatives are constants, and its third and higher derivatives are zeros. Therefore, differentiating equation (1.19) with respect to x_k for $k = 1,2,3,.....,M+N+1$, equation (1.20) is obtained:

$$\frac{\partial E(x+\Delta x)}{\partial x_k} = \frac{\partial E(x)}{\partial x_k} + \sum_{i=1}^{M+N+1} \frac{\partial^2 E}{\partial x_i \partial x_k}\Delta x_i \quad (1.20)$$

According to the necessary condition equation (1.17), for obtaining a minimum, we have

$$\frac{\partial E(x+\Delta x)}{\partial x_k} = \frac{\partial E(x)}{\partial x_k} + \sum_{i=1}^{M+N+1} \frac{\partial^2 E}{\partial x_i \partial x_k}\Delta x_i = 0 \quad (1.21)$$

This equation can be expressed in gradient vector g and Hessian matrix H as

$$g_{(k)} = -H_{(k)}\Delta x$$

where $\Delta x = x_{k+1} - x_k$ and $g_{(k)}$, $H_{(k)}$ are the values of g and H at the k_{th} iteration, respectively. Then,

$$x_{k+1} = x_k - H_{(k)}^{-1} g_{(k)} \tag{1.22}$$

This is Newton's method. The Newton's method maybe rapidly convergent when H_k is non-singular and $E(x)$ is a quadratic function. The main disadvantages of Newton's method are:

- Both the first and second partial derivatives of $E(x)$ must be computed at each iteration in order to construct the gradient and Hessian matrix.
- $H_{(k)}$ maybe singular. It is difficult to compute $H_{(k)}^{-1}$,or even $H_{(k)}^{-1}$ does not exit.
- The Newton's method may not be convergent. i.e., $E(x_{k+1}) > E(x_k)$.

Because of these disadvantages above, the Newton's method is not used to design IIR filter.

1.2.2 Gauss-Newton method and its modifications

Considering equation (1.13), (1.14), (1.15), $E(x)$ should be minimized.

$$E(x) = \sum_{i=1}^{L} f_i^{\,2}(x) \qquad \text{where} \quad f_i(x) = \left| H_i(x) - H_d(e^{j\omega_i}) \right|$$

The first order derivatives of $E(x)$ can be obtained

$$\frac{\partial E(x)}{\partial x_k} = 2\sum_{i=1}^{L} f_i(x)\frac{\partial f_i(x)}{\partial x_k} \qquad \text{(k=1,2,3,……,M+N+1)} \quad (1.23)$$

and the second derivatives can be obtained

$$\frac{\partial^2 E(x)}{\partial x_k \partial x_j} = 2\sum_{i=1}^{L}[\frac{\partial f_i(x)}{\partial x_k}\frac{\partial f_i(x)}{\partial x_j} + f_i(x)\frac{\partial^2 f_i(x)}{\partial x_k \partial x_j}] \quad (1.24)$$

$$(k = 1,2,3,.....,M+N+1;\ j = 1,2,3,.....,M+N+1).$$

If x is not far away from the optimal value, $f_i(x)$ is small, thus it can be assumed that the second order terms on the right hand side of equation (1.24) is small, and the second terms can be ignored.

Thus

$$\frac{\partial^2 E}{\partial x_k \partial x_j} = 2\sum_{i=1}^{L}\frac{\partial f_i(x)}{\partial x_k}\frac{\partial f_i(x)}{\partial x_j} \quad (1.25)$$

Now, a matrix J is defined as following:

$$J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & ..... & \frac{\partial f_1}{\partial x_{M+N+1}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & ..... & \frac{\partial f_2}{\partial x_{M+N+1}} \\ \vdots & \vdots & ..... & \vdots \\ \vdots & \vdots & ..... & \vdots \\ \frac{\partial f_L}{\partial x_1} & \frac{\partial f_L}{\partial x_2} & ..... & \frac{\partial f_L}{\partial x_{M+N+1}} \end{pmatrix} \quad (1.26)$$

J is the Jacobi matrix. Therefore, equation (1.23) can be represented in matrix form Gradient ∇E

$$\nabla E_{(k)} = 2J_{(k)}{}^{H} f_{(k)} \tag{1.27}$$

where $f = (f_1, f_2, f_3, f_4, ..., f_L)^T$, and $J_{(k)}{}^{H}$ is the conjugate transpose of $J_{(k)}$.
Then equation (1.25) can be represented as Hessian matrix

$$H_{(k)} = 2J_{(k)}{}^{H} J_{(k)} \tag{1.28}$$

From the Newton's method, Equation (1.29) can be obtained.

$$H_{(k)}(x_{k+1} - x_k) = -\nabla E_{(k)} \tag{1.29}$$

That is

$$J_{(k)}{}^{H} J_{(k)}(x_{k+1} - x_k) = -J_{(k)}{}^{H} f_{(k)}$$

then

$$x_{k+1} = x_k - (J_{(k)}{}^{H} J_{(k)})^{-1} J_{(k)}{}^{H} f_{(k)} \tag{1.30}$$

where J_k is the value of matrix J in the k_{th} iteration, and $f_{(k)}$ is the value of f in the k_{th} iteration, $J_{(k)}{}^{H}$ is the conjugate transpose of $J_{(k)}$. It is not necessary to explicitly compute $(J_{(k)}{}^{H} J_{(k)})^{-1}$, x_{k+1} can be solved by using *Matlab* left division " \ " operation. It is a least squares sense operation. Then,

$$x_{k+1} = x_k - (J_{(k)}{}^H J_{(k)}) \backslash J_{(k)}{}^H f_{(k)} \tag{1.31}$$

Equation (1.30) is the Gauss-Newton method [Ne-Zheng Sun, (1994)]. In order to obtain real coefficients, $J_{(k)}{}^H f_{(k)}$ should be replaced with the real component of $J_{(k)}{}^H f_{(k)}$ in equation (1.31), while $J_{(k)}{}^H J_{(k)}$ is already a real number. Then,

$$x_{k+1} = x_k - (J_{(k)}{}^H J_{(k)}) \backslash \mathrm{Re}(J_{(k)}{}^H f_{(k)}) \tag{1.32}$$

Like Newton's method, equation (1.32) also has some drawbacks:

- $(J_{(k)}{}^H J_{(k)})$ maybe singular, the result of x_{k+1} maybe far away from the local optimal value.
- The search sequence may not be convergent, i.e., $E(x_{k+1}) > E(x_k)$.

To avoid these difficulties, some modifications should be done on equation (1.32).

- Modification 1:

A positive scalar α_k is introduced to modify equation (1.32). Hence,

$$x_{k+1} = x_k - \alpha_k (J_{(k)}{}^H J_{(k)}) \backslash \mathrm{Re}(J_{(k)}{}^H f_{(k)}) \qquad (0 < \alpha_k \le 1) \tag{1.33}$$

where $(J_{(k)}{}^H J_{(k)}) \backslash J_{(k)}{}^H f_{(k)}$ is the Gauss-Newton direction at the k_{th} iteration, and α_k is the optimal step size along this direction. Thus by properly introducing α_k, inequality $E(x_{k+1}) \le E(x_k)$ can always be guaranteed.

- Modification 2:

A positive scalar λ_k is introduced to modify equation (1.33). Hence,

$$x_{k+1} = x_k - \alpha_k (J_{(k)}{}^H J_{(k)} + \lambda_k \mathrm{I}) \backslash \mathrm{Re}(J_{(k)}{}^H f_{(k)}) \qquad (0 < \alpha_k \le 1) \qquad (1.34)$$

where I is the unit matrix . When $\lambda_k = 0$, equation (1.34) becomes the damped Gauss-Newton method. When λ_k is close to infinity, Δx_k is close to zero. Therefore, $E(x_{k+1}) \le E(x_k)$ can always be guaranteed by increasing the value of λ_k. When λ_k is close to infinity, Modification 2 is close to the steepest descent algorithm [Ne-Zheng Sun, (1999)]. Usually the steepest descent algorithm terminates far from the solution due to round-off effects. Therefore, the steepest descent algorithm is usually inefficient and unreliable in practice [R. Fletcher, (1987)]. We call equation (1.34) as damped Levenberg-Marquardt method. It is different from the Levenberg-Marquardt method:

$$x_{k+1} = x_k - (J_{(k)}{}^H J_{(k)} + \lambda_k \mathrm{I}) \backslash \mathrm{Re}(J_{(k)}{}^H f_{(k)})$$

The above equation is the Levenberg-Marquardt method.

Several design examples show that Modification 1 is superior to Modification 2, and Modification 2 is superior to Gauss-Newton method.

1.3 Least squares designs for the arbitrary magnitude response approximation of IIR filters

1.3.1 Proposed algorithm 1

The least squares error for magnitude approximation is defined as

$$E_1 = \sum_{i=1}^{L} (|H(e^{j\omega_i})| - |H_d(e^{j\omega_i})|)^2 \qquad (1.35)$$

Through solving equation(1.32), (1.33), (1.34), the filter coefficients can be obtained respectively. The designed filter should be stable, i.e., the poles of the filter should be inside the unit circle. For magnitude approximation, this problem can be resolved

through the following procedures: After the iterations, the coefficients is obtained. The denominator coefficients a are stabilized, i.e., the poles outside the unit circle are inflected into the unit circle. This inflection does not affect the filter magnitude frequency response, but does affect the filter phase frequency response. Through this inflection, the coefficients of the filter which magnitude response is approximated to the desired magnitude response can be obtained. Equation (1.35) i.e., the magnitude error is minimized.

In this section, Gauss-Newton method, Modification 1, Modification 2 (damped Levenberg-Marquardt method) are implemented in *Matlab*[6.1]. Here only one algorithm based on Modification 1 is proposed, which is called as *Proposed algorithm 1*. The *Proposed algorithm 1* approximates the magnitude response.

Proposed algorithm 1 works as follows:

a) Choose an initial guess x_1 (Section 1.1 least squares linear equation error IIR filters design algorithm is used to choose x_1).

   Set $k = 1, tol = 0.01$.

b) Compute $E(x_k)$ from (1.14).

c) Compute $\nabla E_{(k)}$ from (1.27). Compute $H_{(k)}$ from (1.28).

d) Set $\alpha_{(k)} = 1$, solve x_{k+1} from (1.33).

e) Compute $E(x_{k+1})$ from (1.14).

f) while $E(x_{k+1}) > E(x_k)$

   $\alpha_k = \alpha_k / 2$.

   Compute x_{k+1} again from (1.33).

   Compute $E(x_{k+1})$ from (1.14).

endwhile.

g) if $\left\| \frac{x_{k+1} - x_k}{\alpha_k} \right\|_2 < tol$

Stop.

else set $k = k + 1$,

Go back to step (b).

Begin another iteration.

endif

h) Stabilize x_{k+1}.

1.3.2 Examples

In this section, several design examples are applied to compare Gauss-Newton method, *proposed algorithm 1*, Modification 2 (damped Levenberg-Marquardt method), a linear programming method used in [APPANNA T.CHOTTERA, GRAHAM A.JULLIEN, (1982)], and Steiglitz-Mcbride method used in [L.E. McBride, H. W. SCHAEFGEN, K. STEIGLITZ, (1966)]. A linear programming technique was used in [APPANNA T.CHOTTERA, GRAHAM A.JULLIEN, (1982)] to design IIR filters for approximating simultaneously the given magnitude and linear phase characteristics. Steiglitz-Mcbride method is a linear iteration method. It includes frequency domain method and time domain method. IIR filters can be designed using Steiglitz-Mcbride method by approximating the magnitude response.

- Example 1.1: Butterworth low-pass filter

The desired filter is a fourth order Butterworth low-pass filter with normalized cutoff frequency 0.4. From Matlab function [b, a]=Butter(4, 0.4), where 4 is the desired butterworth filter order, 0.4 is the cutoff frequency, the desired filter coefficients can be obtained.

The desired filter coefficients are:

Numerators coefficients $b = 4.6583\times10^{-2}, 1.8633\times10^{-1}, 2.7950\times10^{-1}, 1.8633\times10^{-1}, 4.6583\times10^{-2}$.

Denominator coefficients $a = 1.0000, -7.8210\times10^{-1}, 6.7998\times10^{-1}, -1.8268\times10^{-1}, 3.0119\times10^{-2}$.

From Matlab function $[H_d(\omega)]$=freqz(b, a, 10), the desired filter frequency response is obtained. The samples of the desired filter frequency response $H_d(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$H_d(\omega) = 1.0000,$

$8.4000\times10^{-1} - 5.4258\times10^{-1}\,i,$

$3.5659\times10^{-1} - 9.3341\times10^{-1}\,i,$

$-4.1313\times10^{-1} - 8.7980\times10^{-1}\,i,$

$-7.0711\times10^{-1} - 1.9892\times10^{-16}\,i,$

$-1.3538\times10^{-1} + 2.3177\times10^{-1}\,i,$

$9.7474\times10^{-3} + 7.6791\times10^{-2}\,i,$

$1.0342\times10^{-2} + 1.5672\times10^{-2}\,i,$

$2.5243\times10^{-3} + 1.8090\times10^{-3}\,i,$

$1.6745\times10^{-4} + 5.2028\times10^{-5}\,i.$

The samples of the desired filter frequency response magnitude $|H_d(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$|H_d(\omega)|$=1.0000, 1.0000, 9.9920×10^{-1}, 9.7197×10^{-1}, 7.0711×10^{-1}, 2.6842×10^{-1}, 7.7408×10^{-2}, 1.8777×10^{-2}, 3.1056×10^{-3}, 1.7535×10^{-4}.

Let the proposed filter order $M = N = 4$, frequency ω is equally sampled at 10 frequency points ω_i, i.e., $L = 10$. Using *Proposed algorithm 1*, the proposed filter is designed. The proposed filter coefficients are:

Numerators coefficients b =4.6583×10^{-2}, 1.8633×10^{-1}, 2.7950×10^{-1}, 1.8633×10^{-1}, 4.6583×10^{-2}.

Denominator coefficients a =1.0000, -7.8210×10^{-1}, 6.7998×10^{-1}, -1.8268×10^{-1}, 3.0119×10^{-2}.

The samples of the proposed filter frequency response $H(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$H(\omega)$=1.0000,

$8.4000\times10^{-1}-5.4258\times10^{-1}$i,

$3.5659\times10^{-1}-9.3341\times10^{-1}$i,

$-4.1313\times10^{-1}-8.7980\times10^{-1}$i,

$-7.0711\times10^{-1}-1.9892\times10^{-16}$i,

$-1.3538\times10^{-1}+2.3177\times10^{-1}$i,

$9.7474\times10^{-3}+7.6791\times10^{-2}$i,

$1.0342\times10^{-2}+1.5672\times10^{-2}$i,

$2.5243\times10^{-3}+1.8090\times10^{-3}$i,

$1.6745\times10^{-4}+5.2028\times10^{-5}$i.

The samples of the proposed filter frequency response magnitude $|H(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$|H(\omega)|$ =1.0000, 1.0000, 9.9920×10^{-1}, 9.7197×10^{-1}, 7.0711×10^{-1}, 2.6842×10^{-1}, 7.7408×10^{-2}, 1.8777×10^{-2}, 3.1056×10^{-3}, 1.7535×10^{-4}.

The magnitude error of the proposed filter is

E_1=1.0454×10^{-28}.

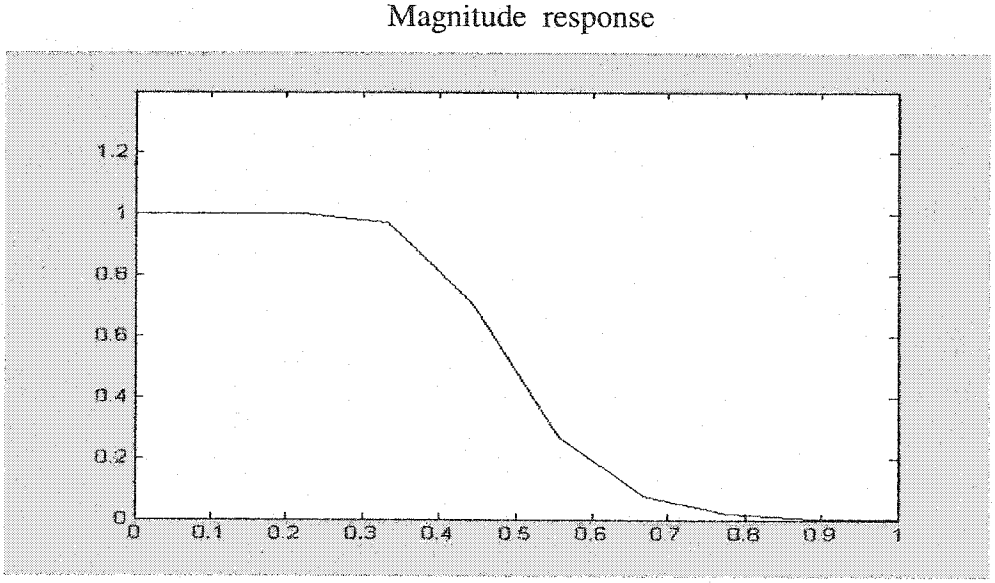


Figure 1.1: Magnitude of the desired filter (Example 1.1)

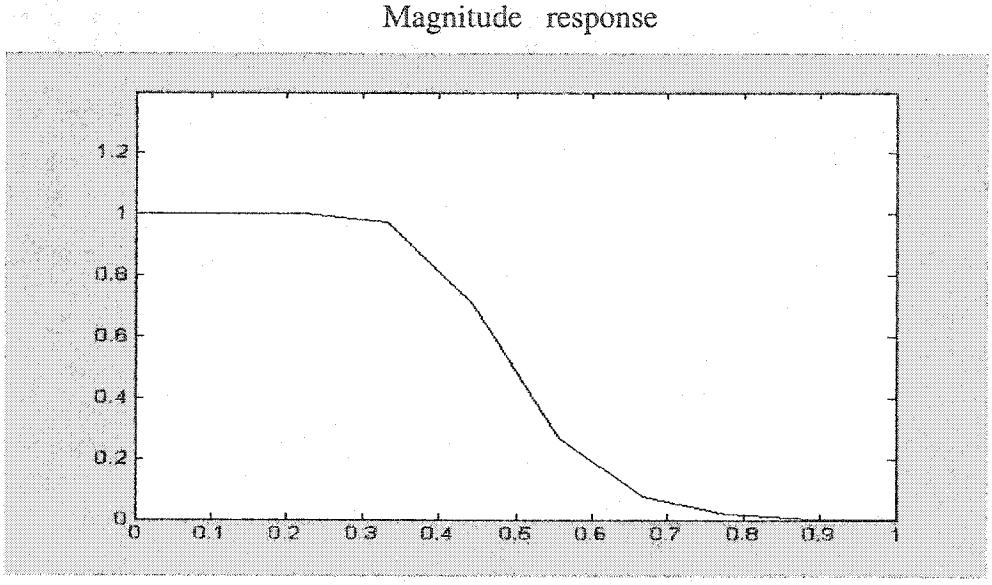


Figure 1.2: Magnitude of the proposed filter (Example 1.1)

- Example 1.2: low-pass filter

The desired filter frequency response is as follows [APPANNA T.CHOTTERA, GRAHAM A.JULLIEN, (1982)]:

$$H_d(e^{j\omega}) = \begin{cases} 1.0, & 0 \le \omega \le 0.5\pi \\ e^{-186.6(0.5\pi-\omega)^2}, & 0.5\pi < \omega \le \pi \end{cases}$$

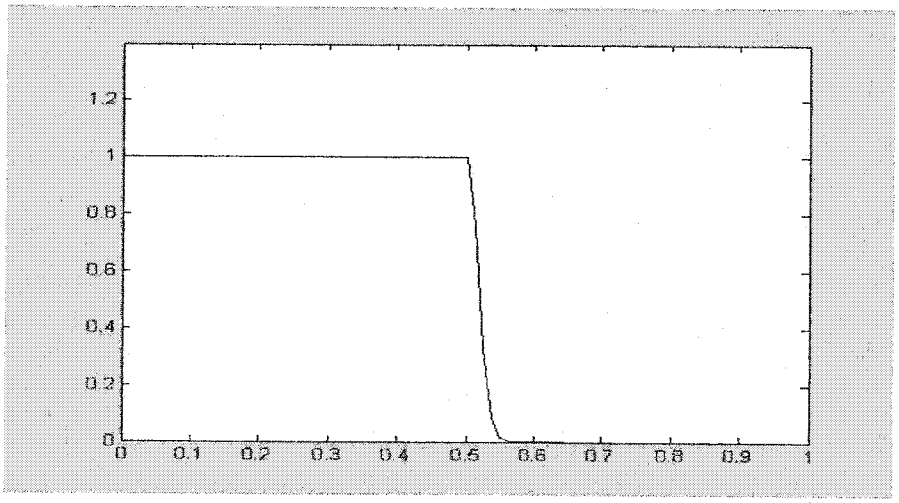
Let the proposed filter frequency sampled points $L = 81$ between $0 \sim \pi$. The order of the proposed filter $M = N = 18$. By using the *Proposed algorithm 1* , the proposed filter is designed.

The proposed filter coefficients are:

Numerator coefficients $b = 5.5882\times10^{-3}, 4.6127\times10^{-2}, 2.0940\times10^{-1}, 6.7129\times10^{-1}, 1.6745, 3.4173, 5.8743, 8.6610, 1.1075\times10^{1}, 1.2360\times10^{1}, 1.2066\times10^{1}, 1.0293\times10^{1}, 7.6337, 4.8754, 2.6394, 1.1794, 4.1651\times10^{-1}, 1.0603\times10^{-1}, 1.6054\times10^{-2}$.

Denominator coefficients $a = 1.0000, 9.9684\times10^{-1}, 5.2986, 4.6912, 1.1875\times10^{1}, 9.2416, 1.4611\times10^{1}, 9.8513, 1.0703\times10^{1}, 6.1252, 4.7285, 2.2264, 1.2132, 4.4674\times10^{-1}, 1.6328\times10^{-1}, 4.2807\times10^{-2}, 9.2283\times10^{-3}, 1.3929\times10^{-3}, 1.4176\times10^{-4}$.

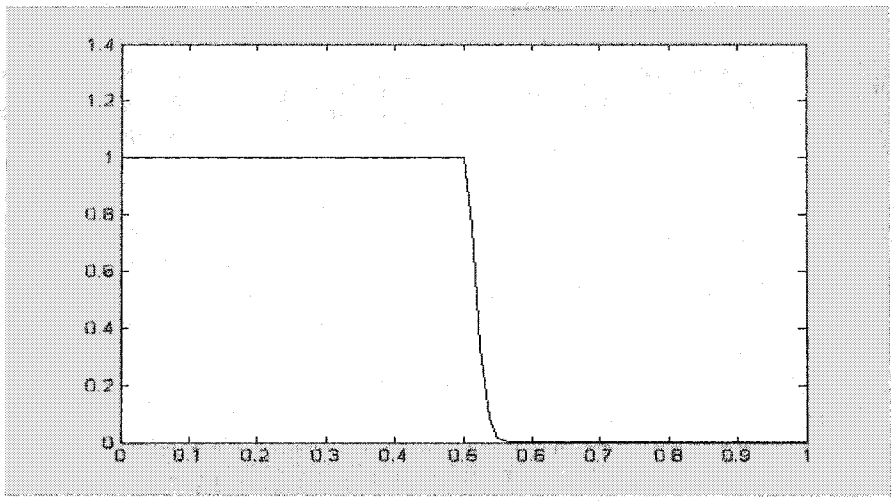
Magnitude response



ω/π

Figure 1.3: Magnitude of the desired filter (Example 1.2)

Magnitude response



ω/π

Figure 1.4: Magnitude of the proposed filter (Example 1.2)

The comparison results of the magnitude error E_1 are shown in table 1.1.

Table 1.1 Magnitude error comparison results of example 1.2

	Gauss-Newton method	Proposed algorithm 1	Damped L-M method	Paper [A.T.Chottrea (1982)]	St-Mc method
E_1	1.2308×10^{-5}	1.5958×10^{-6}	2.9212×10^{-5}	1.2190×10^{-1}	1.7140×10^{-1}

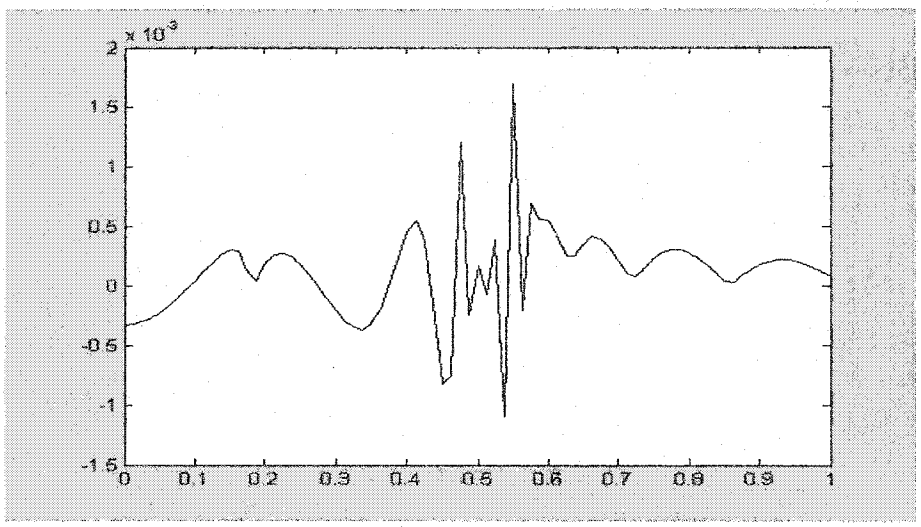


Figure 1.5: Magnitude error of the IIR filter (Gauss-Newton method)

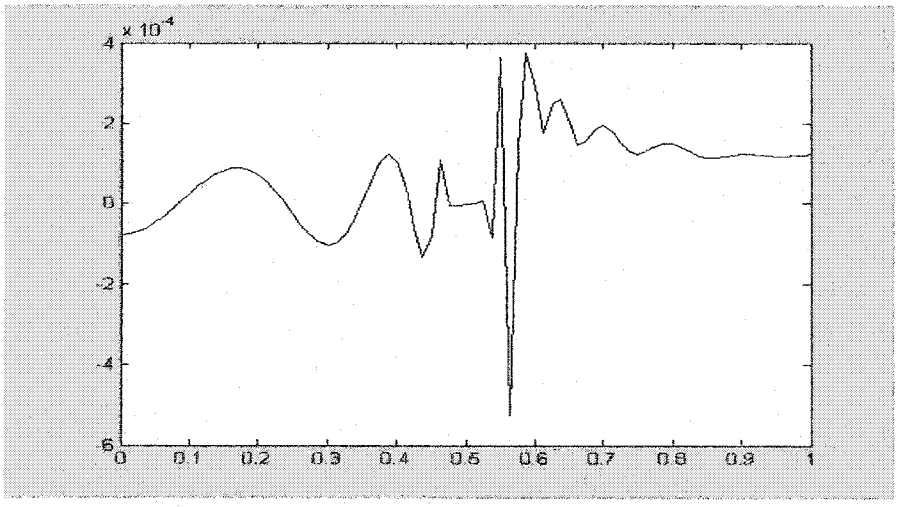


Figure 1.6: Magnitude error of the IIR filter (Proposed algorithm 1)

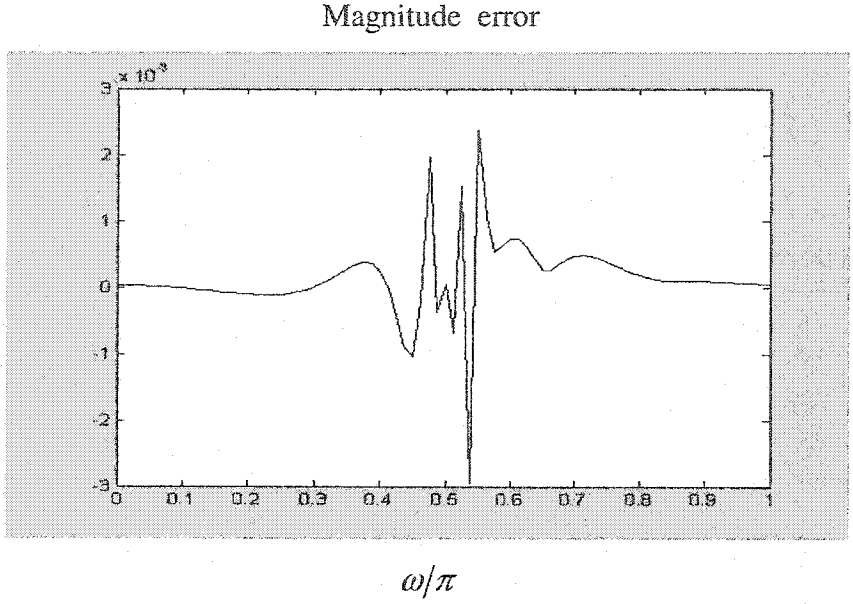


Figure 1.7: Magnitude error of the IIR filter (damped Levenberg-Marquardt method)

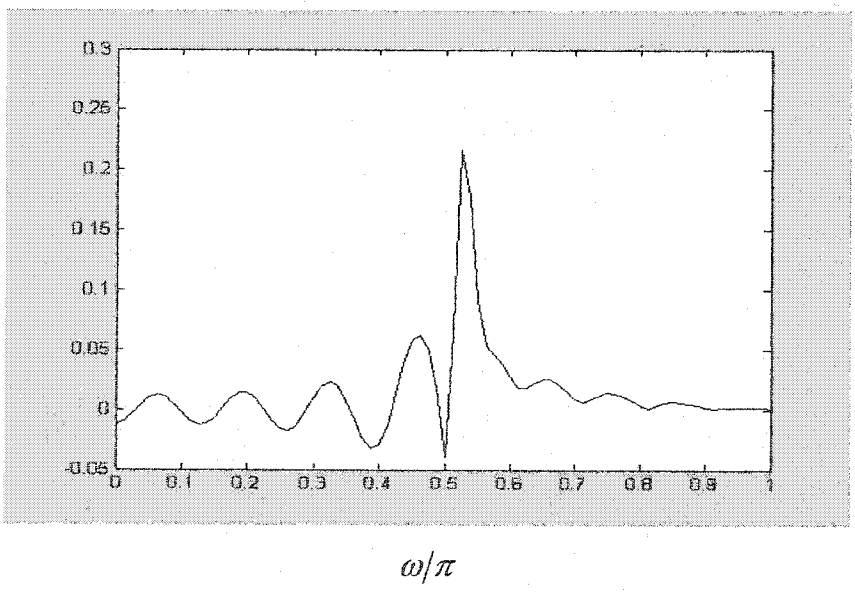


Figure 1.8: Magnitude error of the IIR filter (paper [A.T.Chottera,(1982)])

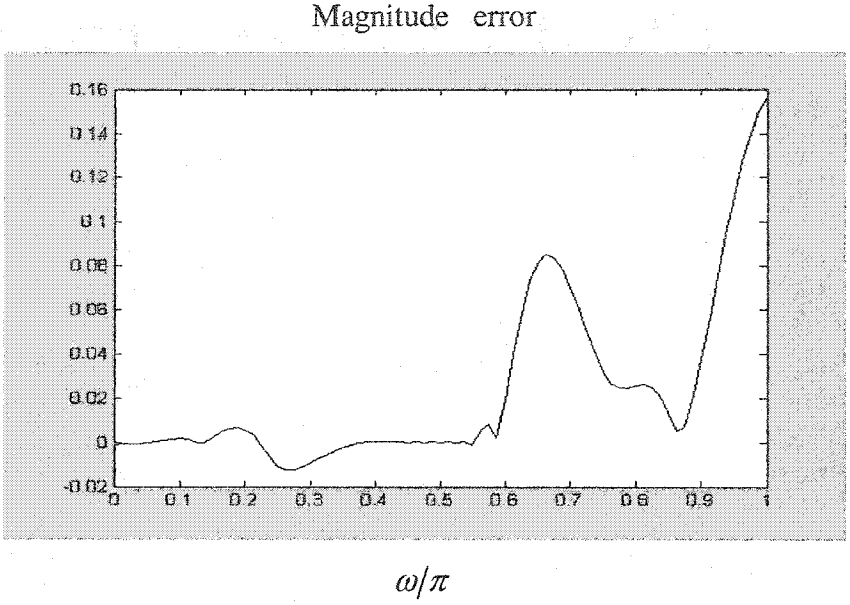


Figure 1.9: Magnitude error of the IIR filter (St-Mc method)

- Example 1.3: a five-band filter

The desired specifications are for a five-band filter, with three stop bands and two pass bands. The desired specifications are as follows [APPANNA T.CHOTTERA, GRAHAM A.JULLIEN, (1982)]:

$$H_d(e^{j\omega}) = e^{-46.6(0.2\pi-\omega)^2}, \qquad 0 \le \omega < 0.2\pi$$

$$H_d(e^{j\omega}) = 1.0, \qquad 0.2\pi \le \omega \le 0.3\pi$$

$$H_d(e^{j\omega}) = e^{-46.6(0.3\pi-\omega)^2}, \qquad 0.3\pi < \omega \le 0.45\pi$$

$$H_d(e^{j\omega}) = e^{-46.6(0.6\pi-\omega)^2}, \qquad 0.45\pi < \omega < 0.6\pi$$

$$H_d(e^{j\omega}) = 1.0, \qquad 0.6\pi \le \omega \le 0.8\pi$$

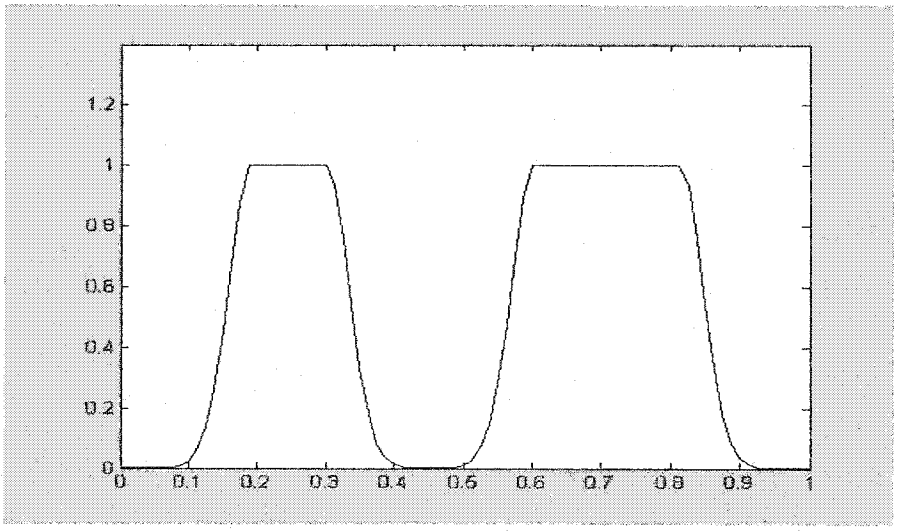
$$H_d(e^{j\omega}) = e^{-46.6(0.8\pi-\omega)^2}, \qquad 0.8\pi < \omega \le \pi$$

Let the proposed filter frequency sampled points $L=81$ between $0\sim\pi$, the order of the proposed filter $M=N=20$. By using the *Proposed algorithm 1*, the proposed filter is designed. The proposed filter coefficients are :

Numerator coefficients b =2.3143×10^{-2}, 4.4894×10^{-2}, 4.8402×10^{-2}, 1.1343×10^{-2}, -1.2774×10^{-2}, -1.1636×10^{-1}, -1.9487×10^{-1}, -1.0938×10^{-2}, -9.7838×10^{-3}, 7.8838×10^{-2}, 3.1170×10^{-1}, 7.8834×10^{-2}, -9.7912×10^{-3}, -1.0936×10^{-2}, 4.4895×10^{-2}, 2.3139×10^{-2}, -1.9488×10^{-1}, -1.1636×10^{-1}, -1.2766×10^{-2}, 1.1340×10^{-2}, 4.8403×10^{-2}.

Denominator coefficients a =1.0000, 6.6410×10^{-1}, 9.6902×10^{-1}, 4.1333×10^{-1}, 1.0477, 6.6397×10^{-1}, 1.1376, 3.3702×10^{-1}, 8.6960×10^{-1}, 5.8004×10^{-1}, 8.7543×10^{-1}, 3.3190×10^{-1}, 4.1789×10^{-1}, 1.8333×10^{-1}, 3.1469×10^{-1}, 1.3191×10^{-1}, 1.3212×10^{-1}, 4.9578×10^{-2}, 8.4613×10^{-2}, 4.4206×10^{-2}, 2.2436×10^{-2}.

Magnitude response



ω/π

Figure 1.10: Magnitude of the desired filter (Example 1.3)

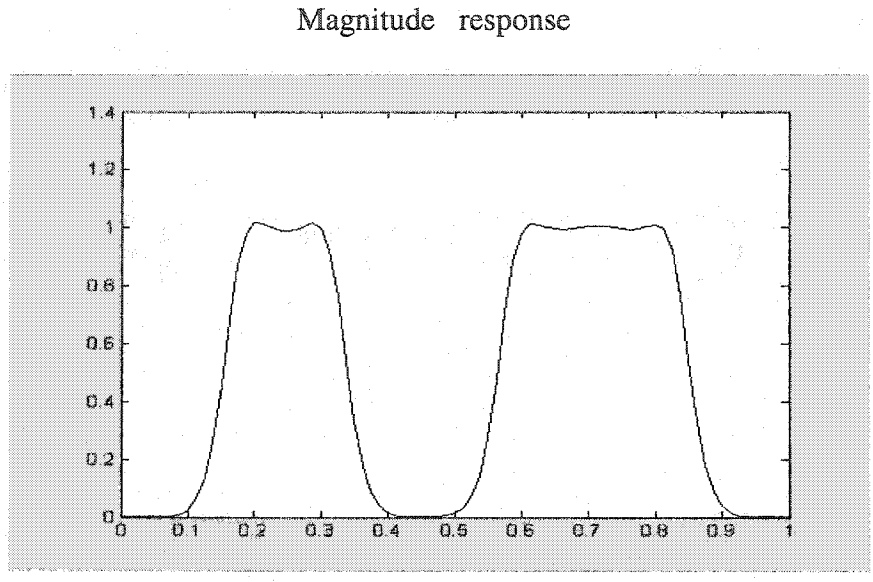


Figure 1.11: Magnitude of the proposed filter (Example 1.3)

The comparison results magnitude error E_1 are shown in table 1.2.

Table 1.2 Magnitude error comparison results of example 1.3

	Gauss-Newton method	Proposed algorithm 1	Damped L-M method	Paper [A.T.Chottera (1982)]	Mc-St method
E_1	3.2000 $\times 10^{-3}$	3.2000 $\times 10^{-3}$	3.3000 $\times 10^{-3}$	2.1430 $\times 10^{-1}$	4.7700 $\times 10^{-2}$

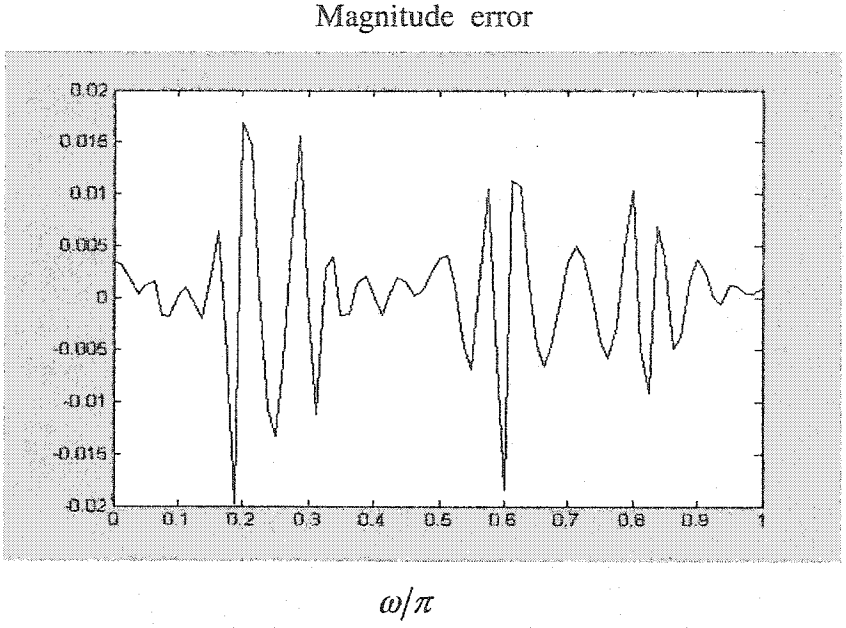


Figure 1.12: Magnitude error of the IIR filter (Gauss-Newton method)

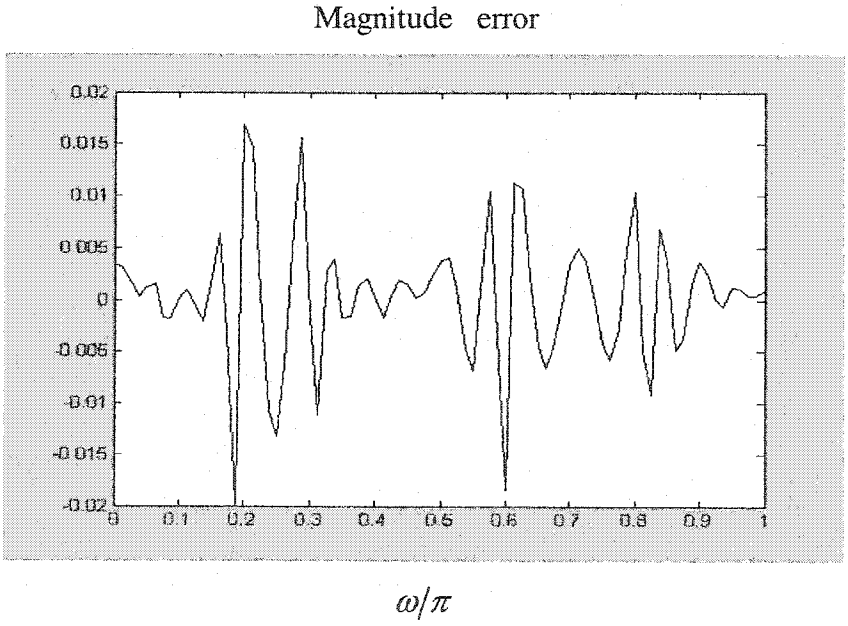


Figure 1.13: Magnitude error of the IIR filter (Proposed algorithm 1)

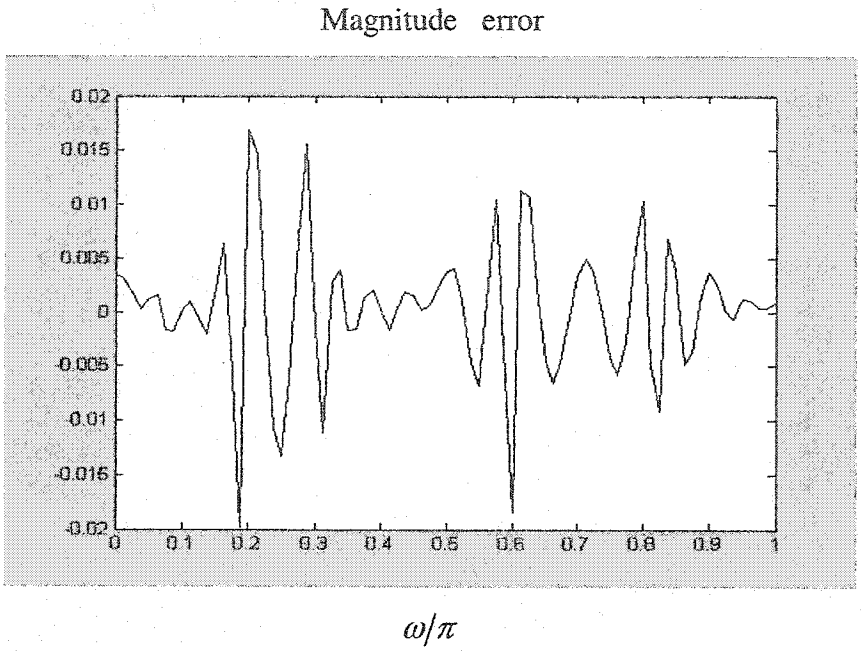


Figure 1.14: Magnitude error of the IIR filter (damped Levenberg-Marquardt method)

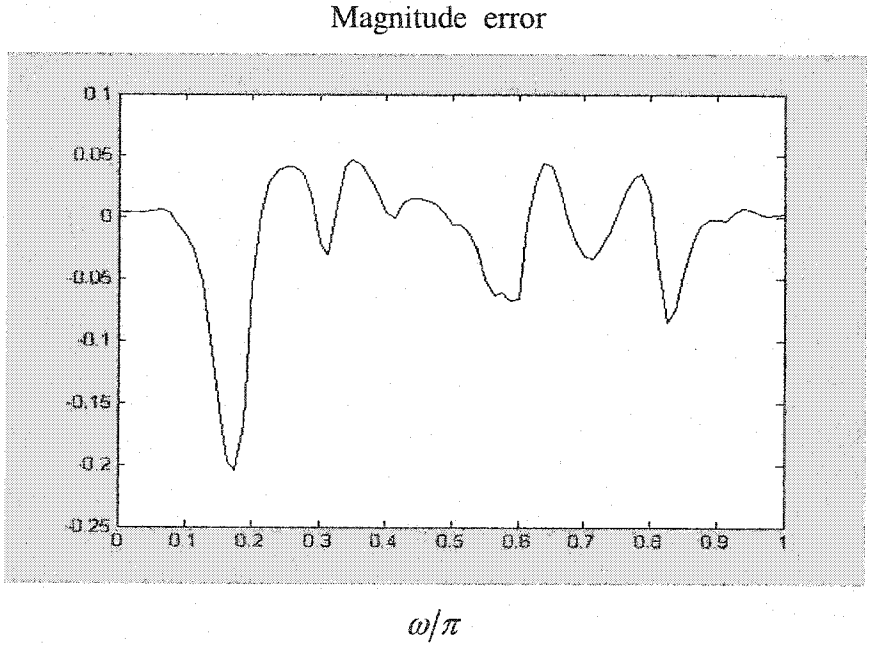


Figure1.15: Magnitude error of the IIR filter (paper [A.T.Chottera,(1982)])

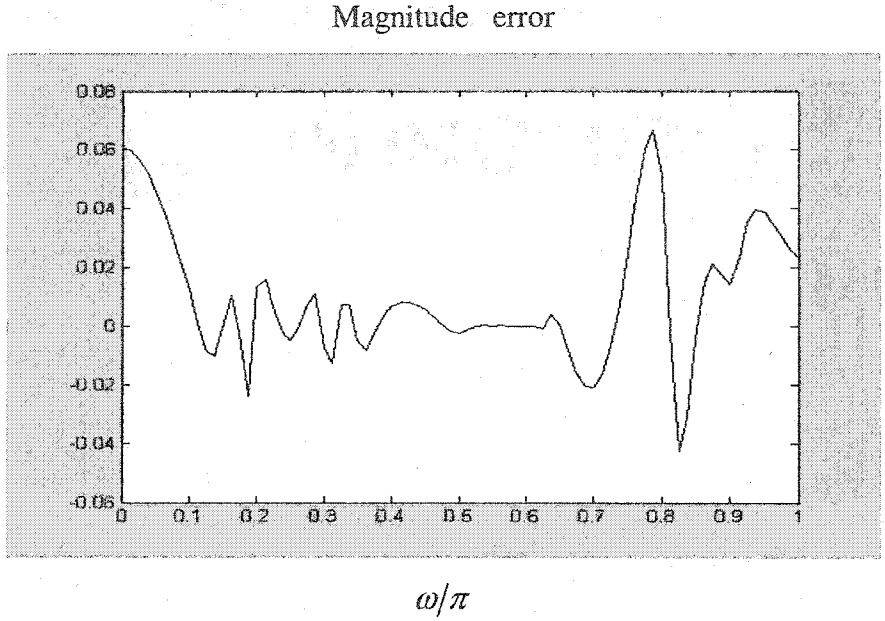


Figure 1.16: Magnitude error of the IIR filter (St-Mc method)

- Example 1.4: differentiator

The magnitude characteristics of the desired differentiator is given by [APPANNA T.CHOTTERA, GRAHAM A.JULLIEN, (1982)]:

$$H_d(e^{j\omega}) = \frac{\omega}{\pi} \qquad 0 \le \omega \le \pi .$$

Let the proposed filter frequency sampled points $L = 81$ between $0 \sim \pi$. The order of the filter $M = N = 17$. By using the *Proposed algorithm 1,* the proposed filter is designed.

The proposed filter coefficients:

Numerator coefficients $b = 7.0028\times10^{-3}$, -2.8998×10^{-2}, -1.0304×10^{-1}, 3.1335×10^{-1}, 3.6020×10^{-1}, -9.4954×10^{-1}, -7.1665×10^{-1}, 1.1466, 1.3246,

-6.2376×10^{-1}, -1.8018, 2.9306×10^{-1}, 1.2933, -2.2635×10^{-1}, -4.0513×10^{-1}, 8.5703×10^{-2}, 3.8100×10^{-2}, -6.7297×10^{-3}.

Denominator coefficients a =1.0000, -6.7987×10^{-2}, -3.1282, -6.4187×10^{-1}, 3.8459, 2.3705, -2.3427, -3.0134, 7.2803×10^{-1}, 1.8362, -1.0416×10^{-1}, -5.6310×10^{-1}, 4.2469×10^{-3}, 8.0634×10^{-2}, 2.4289×10^{-4}, -4.2680×10^{-3}, -7.9547×10^{-6}, 3.7889×10^{-7}.

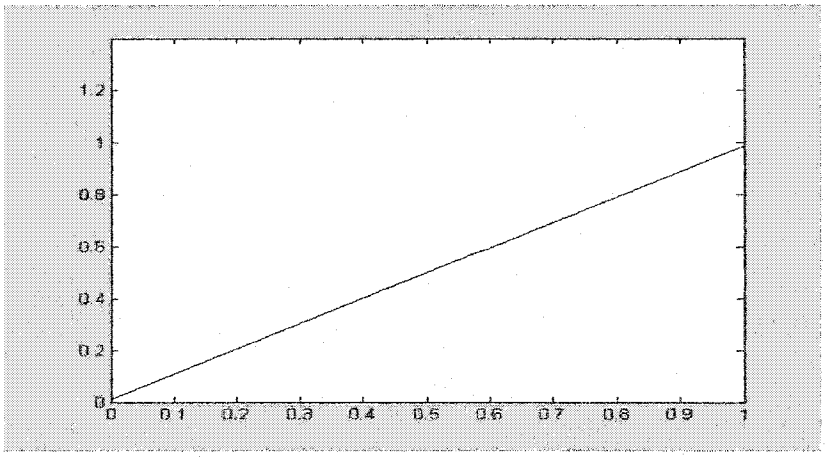


Figure 1.17: Magnitude of the desired filter (Example 1.4)

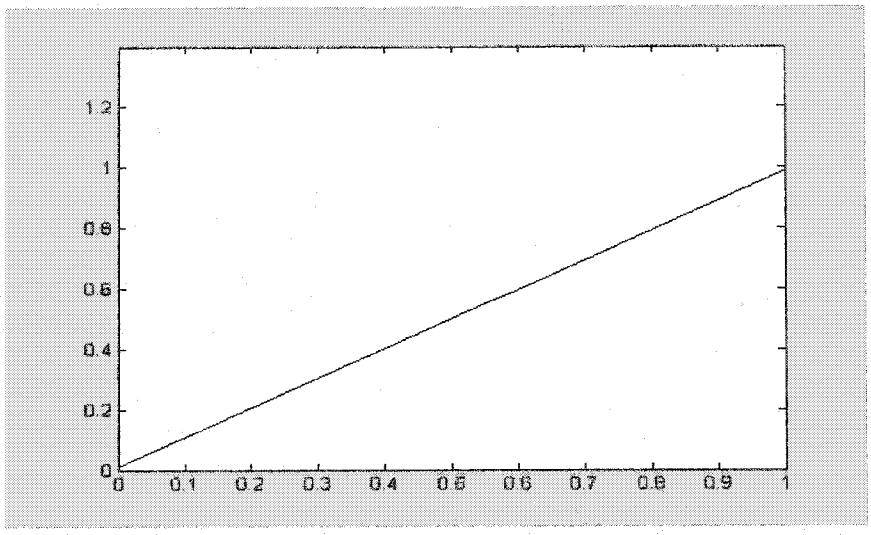


Figure 1.18: Magnitude of the proposed filter (Example 1.4)

The comparison results are shown in table 1.3.

Table 1.3 Magnitude error comparison results of example 1.4

	Gauss-Newton method	Proposed algorithm 1	Damped L-M method	Paper [A.T.Chottera, (1982)]	Mc-St method
E_1	1.5787 $\times 10^{-7}$	8.2808 $\times 10^{-8}$	2.6128 $\times 10^{-7}$	4.3000 $\times 10^{-3}$	2.3300 $\times 10^{-2}$

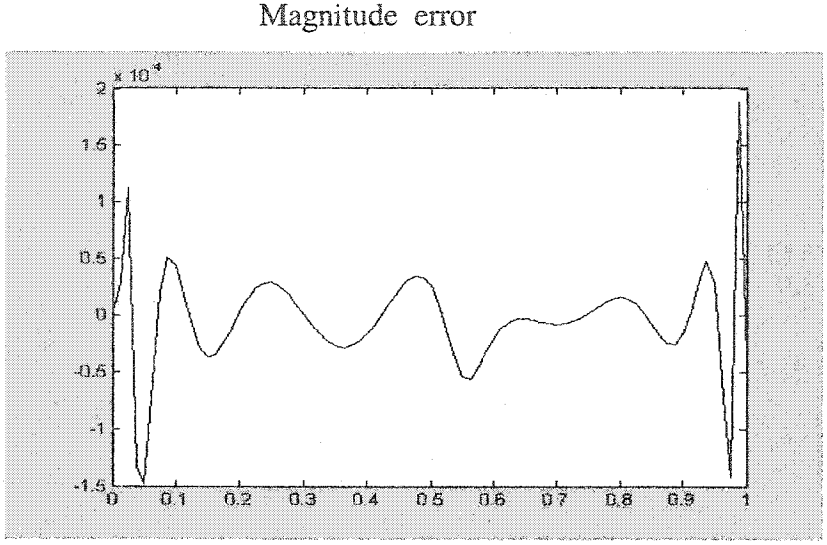


Figure 1.19: Magnitude error of the IIR filter (Gauss-Newton method)

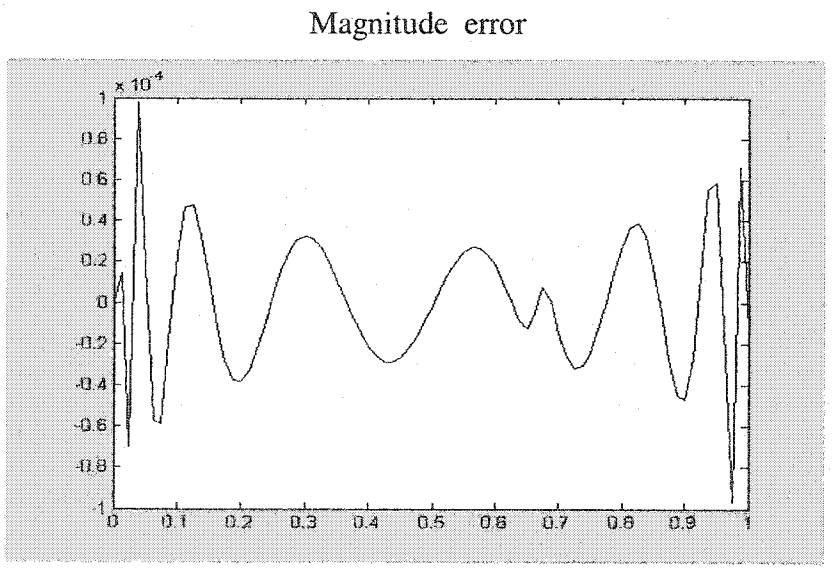


Figure 1.20: Magnitude error of the IIR filter (Proposed algorithm 1)

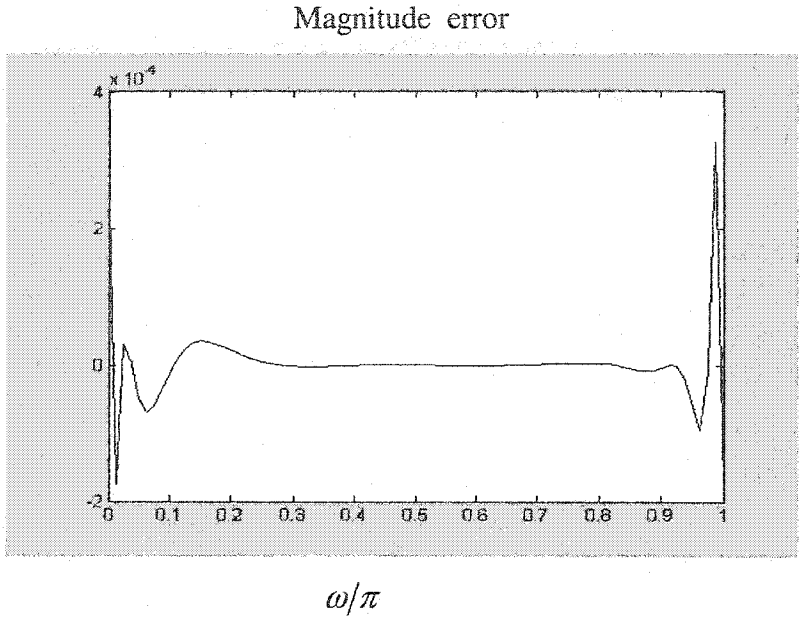


Figure 1.21: Magnitude error of the IIR filter (damped Levenberg-Marquardt method)

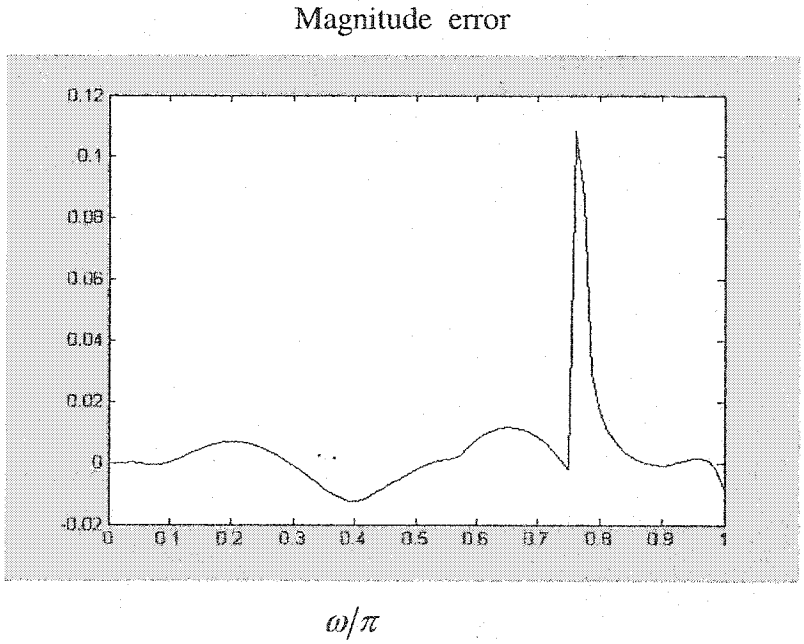


Figure 1.22: Magnitude error of the IIR filter (paper [A.T.Chottera,(1982)])

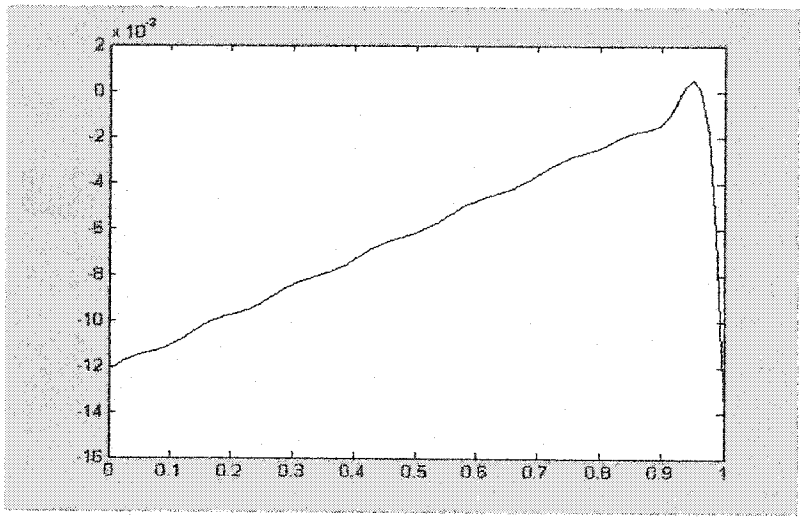


Figure 1.23: Magnitude error of the IIR filter (St-Mc method)

1.4 Least squares design of stable IIR filters with arbitrary magnitude response and phase response and prescribed stability region

1.4.1 Proposed algorithm 2

Through solving (1.32),(1.33),(1.34), the filter coefficients can be obtained respectively. The designed filter should be stable, i.e., the poles of the filter should be inside the unit circle. For approximating simultaneously magnitude response and phase response in the prescribed stability region, this problem is resolved through the following procedures: In each iteration, the coefficients can be obtained. The denominator coefficients a are stabilized, i.e., the poles outside the unit circle are inflected into the unit circle. If some poles radius is larger than the prescribed pole radius, these poles are set on the circle of the prescribed radius. Thus stable filter can be obtained. Not only the magnitude response but also the phase response can be approximated to the desired frequency response. Also the stable region can be arbitrarily prescribed in advance.

The *proposed algorithm 2* works as follows:

a) Choose an initial stable guess x_1 (Section 1.1 least squares linear equation error IIR filters design algorithm is used to choose x_1).

Set $k=1$, $tol=0.01$.

b) Compute $E(x_k)$ from (1.14).

c) Compute $\nabla E_{(k)}$ from (1.27). Compute $H_{(k)}$ from (1.28).

d) Set $\alpha_{(k)}=1$, solve x_{k+1} from (1.33).

Stabilize x_{k+1} in the prescribed stable region.

e) Compute $E(x_{k+1})$ from (1.14).

f) while $E(x_{k+1})>E(x_k)$

$\alpha_k=\alpha_k/2$.

Compute x_{k+1} again from (1.33).

Stabilize x_{k+1} in the prescribed stable region.

Compute $E(x_{k+1})$ from (1.14).

endwhile.

g) if $\left\|\frac{x_{k+1}-x_k}{\alpha_k}\right\|_2 < tol$

Stop. x_{k+1} is the final result.

else set $k=k+1$,

Go back to step (b).

Begin another iteration.

endif

1.4.2 Examples

In this section, several design examples are applied to compare Gauss-Newton method, proposed algorithm 2, damped Levenberg-Marquardt method, an algorithm

proposed by Lu in [Wu-Sheng Lu, (1998)], and an algorithm proposed by Tarczynski in [Andrzej Tarczynski,(1999)].

Lu used linear least squares error criterion and quadratic programming techniques to design stable IIR filters. Tarczynski used the non-linear cost function to design IIR filters. The cost function includes the frequency-domain components and time-domain components. Frequency-domain components are used to make the filter represent desired frequency response, The time domain components make the filter stable.

- Example 1.5

This example is a fourth Chebyshev I low-pass filter, with 0.5 decibel of peak-peak ripple in the passband. The cutoff frequency is 0.6π. From Matlab function [b,a]=Cheby1 (4, 0.5, 0.6), the coefficients b, a of the desired filter can be obtained. From the Matlab function $[H_d(\omega)]$= freqz(b, a,10), the desired filter frequency response is obtained. The samples of the desired filter frequency response $H_d(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$H_d(\omega) = 9.4406\times10^{-1},$$
$$9.0743\times10^{-1} - 2.9572\times10^{-1}\,i,$$
$$7.6910\times10^{-1} - 6.0761\times10^{-1}\,i,$$
$$4.5408\times10^{-1} - 8.9076\times10^{-1}\,i,$$
$$-5.8016\times10^{-2} - 9.7646\times10^{-1}\,i,$$
$$-6.1995\times10^{-1} - 7.1282\times10^{-1}\,i,$$
$$-8.4136\times10^{-1} + 4.2821\times10^{-1}\,i,$$
$$7.1981\times10^{-2} + 1.4147\times10^{-1}\,i,$$
$$1.4834\times10^{-2} + 9.7990\times10^{-3}\,i,$$
$$8.1842\times10^{-4} + 2.2293\times10^{-4}\,i.$$

The samples of the desired filter frequency response magnitude $|H_d(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$|H_d(\omega)| = 9.4406\times10^{-1}, 9.5440\times10^{-1}, 9.8015\times10^{-1}, 9.9982\times10^{-1}, 9.7818\times10^{-1},$$
$$9.4470\times10^{-1}, 9.4406\times10^{-1}, 1.5873\times10^{-1}, 1.7778\times10^{-2}, 8.4824\times10^{-4}.$$

The desired filter coefficients:

Numerator coefficients b= $1.2927\times10^{-1}, 5.1708\times10^{-1}, 7.7562\times10^{-1}, 5.1708\times10^{-1}, 1.2927\times10^{-1}$.

Denominator coefficients a =1.0000, $3.5159\times10^{-1}, 7.7075\times10^{-1}, -6.1412\times10^{-2}, 1.2994\times10^{-1}$.

The maximum pole radius of the desired filter is 0.8509. Let the proposed filter order $M = N = 4$, the proposed filter frequency sampled points $L = 10$ between $0\sim\pi$. By using *Proposed algorithm 2,* set the maximum pole radius as 0.85, the proposed filter is designed.

The proposed filter coefficients:

Numerator coefficients b= $1.2927\times10^{-1}, 5.1708\times10^{-1}, 7.7562\times10^{-1}, 5.1708\times10^{-1}, 1.2927\times10^{-1}$.

Denominator coefficients a =1.0000, $3.5159\times10^{-1}, 7.7075\times10^{-1}, -6.1412\times10^{-2}, 1.2994\times10^{-1}$.

The samples of the proposed filter frequency response $H(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$H(\omega) = 9.4490\times10^{-1}$,

9.0816×10^{-1}-2.9620×10^{-1}i,

7.6945×10^{-1}-6.0852×10^{-1}i,

4.5367×10^{-1}-8.9186×10^{-1}i,

-5.9417×10^{-2}-9.7697×10^{-1}i,

-6.2160×10^{-1}-7.1119×10^{-1}i,

-8.3652×10^{-1}+4.2724×10^{-1}i,

7.1593×10^{-2}+1.4156×10^{-1}i,

1.4830×10^{-2}+9.8212×10^{-3}i,

8.1891×10^{-4}+2.2349×10^{-4}i.

The samples of the desired filter frequency response magnitude $|H(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$|H(\omega)| = 9.4490\times10^{-1},\ 9.5524\times10^{-1},\ 9.8099\times10^{-1},\ 1.0006,\ 9.7878\times10^{-1},$$
$$9.4455\times10^{-1},\ 9.3931\times10^{-1},\ 1.5863\times10^{-1},\ 1.7787\times10^{-2},\ 8.4886\times10^{-4}.$$

In order to show that the proposed filter maximum pole radius can be set arbitrarily, the proposed filter poles radius are given:

$$8.5000\times10^{-1},\ 8.5000\times10^{-1},\ 4.2366\times10^{-1},\ 4.2366\times10^{-1}.$$

The proposed filter frequency response error:

$$E(x) = 3.5980\times10^{-5}.$$

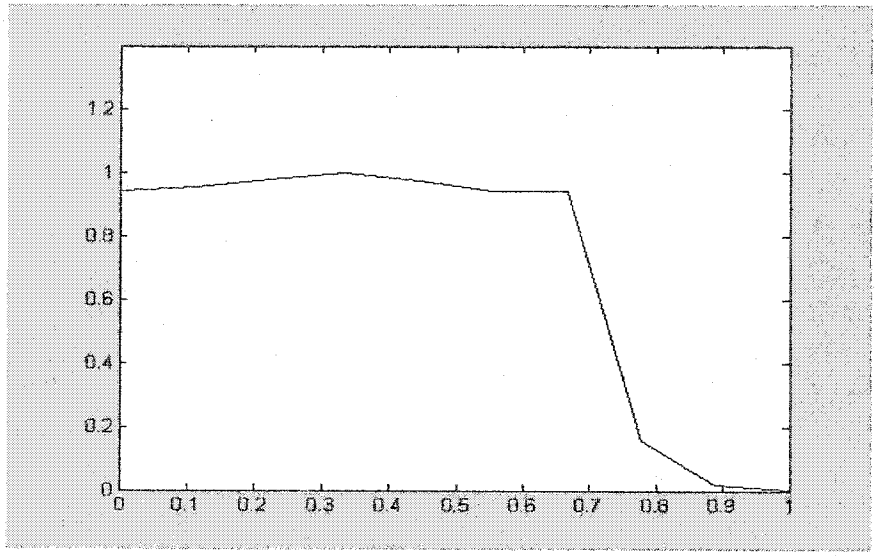


Figure 1.24: Magnitude of the desired filter (Example 1.5)

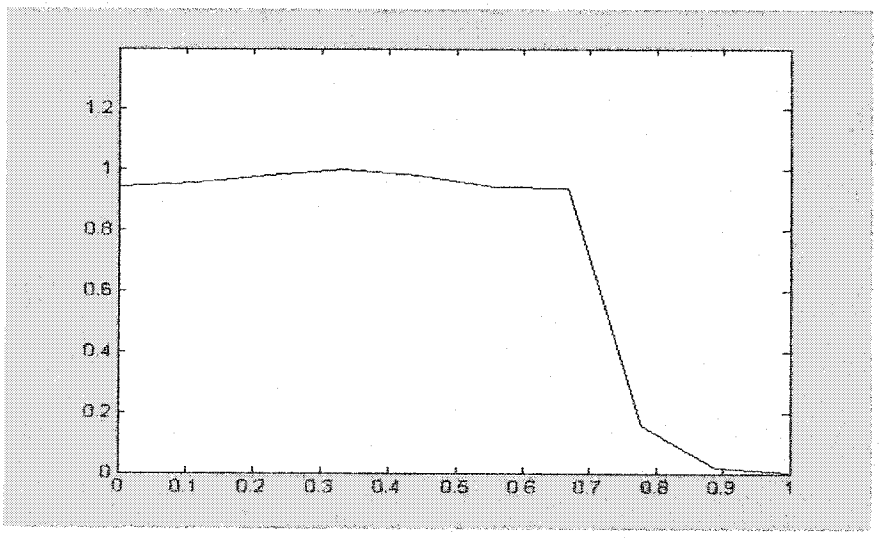


Figure 1.25: Magnitude of the proposed filter (Example 1.5)

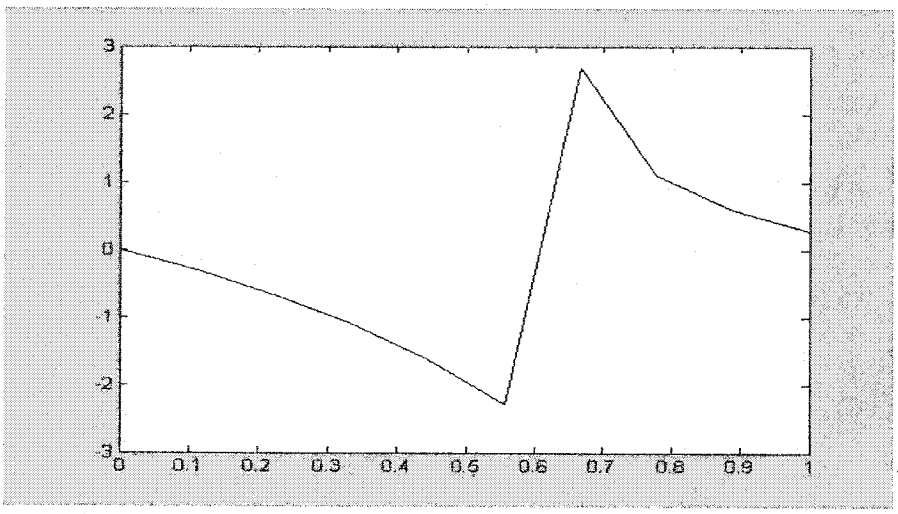


Figure 1.26: Phase of the desired filter (Example 1.5)

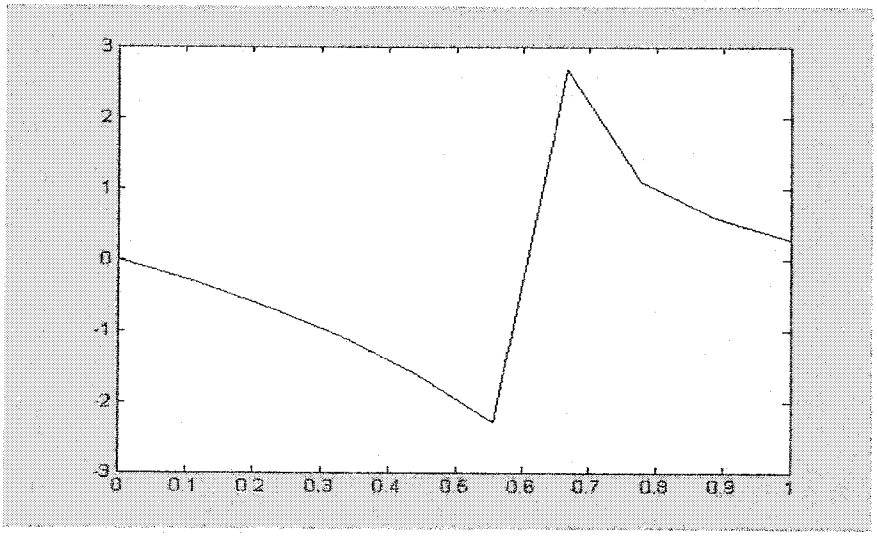


Figure 1.27: Phase of the proposed filter (Example 1.5)

- Example 1.6

The desired frequency [Lu, (1998), A. Tarczynski, (1999)] is given by:

$$H_d(e^{j\omega}) = \begin{cases} e^{-12j\omega} & \omega \geq 0.5\pi \\ 0 & \omega < 0.5\pi \end{cases}$$

The order of this filter in [Lu,(1998)] and [A.Tarczynski (1999)] is 14. Let the proposed filter order $M = N = 14$. Let the proposed filter frequency sampled points $L = 256$ between $0 \sim \pi$. The coefficients of this filter have been published in [Lu,(1998)]. The maximum pole radius of this filter in [Lu,(1998)] is 0.9276. The coefficients of this filter also have been published in [A. Tarczynski, (1999)]. The maximum pole radius of this filter in [A. Tarczynski, (1999)] is 0.9913. In order to make fair comparison, two comparisons are made. Firstly, the maximum radius of the poles is set as 0.9913. Secondly, the maximum radius of the poles is set as 0.9276.

By using *Proposed algorithm 2*, the proposed filters are designed.

When the proposed filter maximum poles radius is 9.9130×10^{-1}, the proposed filter coefficients are:

Numerator coefficients: $b = 2.0519\times10^{-4}, 3.1824\times10^{-2}, 1.4808\times10^{-1}, 3.8240\times10^{-1},$
$6.8932\times10^{-1}, 9.4706\times10^{-1}, 1.0297, 8.8972\times10^{-1},$
$5.7397\times10^{-1}, 2.2619\times10^{-1}, 1.3165\times10^{-2}, -1.4156\times10^{-1},$
$-1.1365\times10^{-1}, -7.4841\times10^{-2}, -4.1713\times10^{-2}.$

Denominator coefficients: $a = 1.0000, 4.8468, 1.3964\times10^{1}, 2.9308\times10^{1}, 4.8320\times10^{1},$
$6.5143\times10^{1}, 7.3256\times10^{1}, 6.9347\times10^{1}, 5.5325\times10^{1},$

3.6963×10^{1}, 2.0367×10^{1}, 8.9983, 3.0280, 7.0023×10^{-1}, 8.4670×10^{-2}.

The proposed filter poles radius:

9.9130×10^{-1}, 9.9130×10^{-1}, 9.7603×10^{-1}, 9.7603×10^{-1}, 8.8439×10^{-1}, 8.8439×10^{-1}, 7.9725×10^{-1}, 7.9725×10^{-1}, 7.5575×10^{-1}, 7.5575×10^{-1}, 7.4332×10^{-1}, 7.4332×10^{-1}, 7.5928×10^{-1}, 7.5928×10^{-1}.

When the proposed filter maximum poles radius is 9.2760×10^{-1}, the proposed filter coefficients are:

Numerator coefficients: b =2.2765×10^{-5}, 3.1069×10^{-2}, 2.1482×10^{-1}, 7.1470×10^{-1}, 1.5019, 2.2124, 2.4187, 2.0578, 1.4282, 8.1959×10^{-1}, 4.1289×10^{-1}, 2.6395×10^{-1}, 2.2479×10^{-1}, 1.3993×10^{-1}, 3.8350×10^{-2}.

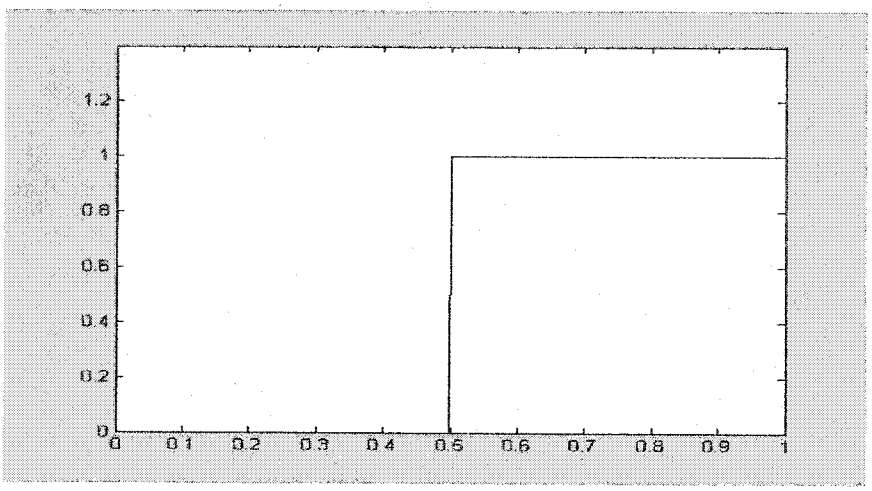
Denominator coefficients: a =1.0000, 7.0567, 2.5019×10^{1}, 5.9157×10^{1}, 1.0414×10^{2}, 1.4414×10^{2}, 1.6139×10^{2}, 1.4816×10^{2}, 1.1186×10^{2}, 6.9029×10^{1}, 3.4249×10^{1}, 1.3244×10^{1}, 3.7748, 7.1031×10^{-1}, 6.6656×10^{-2}.

The proposed filter poles radius:

9.1820×10^{-1}, 9.1820×10^{-1}, 8.1060×10^{-1}, 8.1060×10^{-1}, 7.4982×10^{-1}, 7.4982×10^{-1}, 9.1266×10^{-1}, 9.1266×10^{-1},

$7.4273\times10^{-1}, 7.4273\times10^{-1}, 9.2760\times10^{-1}, 9.1178\times10^{-1},$
$7.4210\times10^{-1}, 7.4210\times10^{-1}.$

Magnitude response



ω/π

Figure 1.28: Magnitude of the desired filter (Example 1.6)

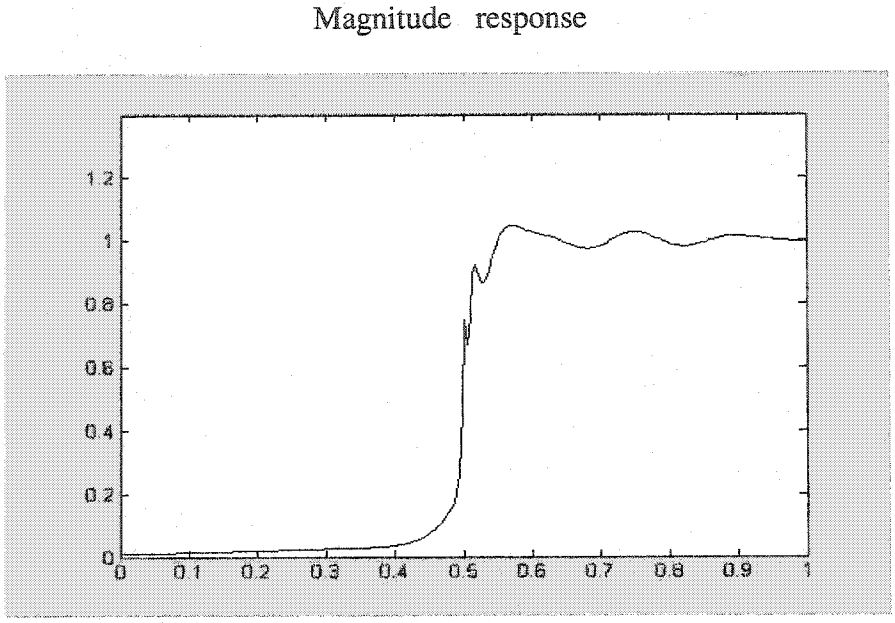


Figure 1.29: Magnitude of the proposed filter (max. pole radius =0.9913, Example 1.6)

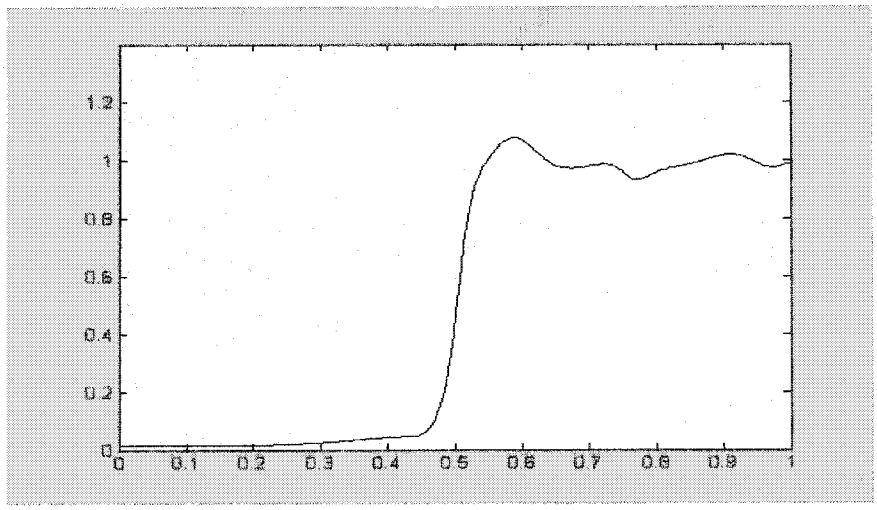


Figure 1.30: Magnitude of the proposed filter (max. pole radius =0.9276, Example 1.6)

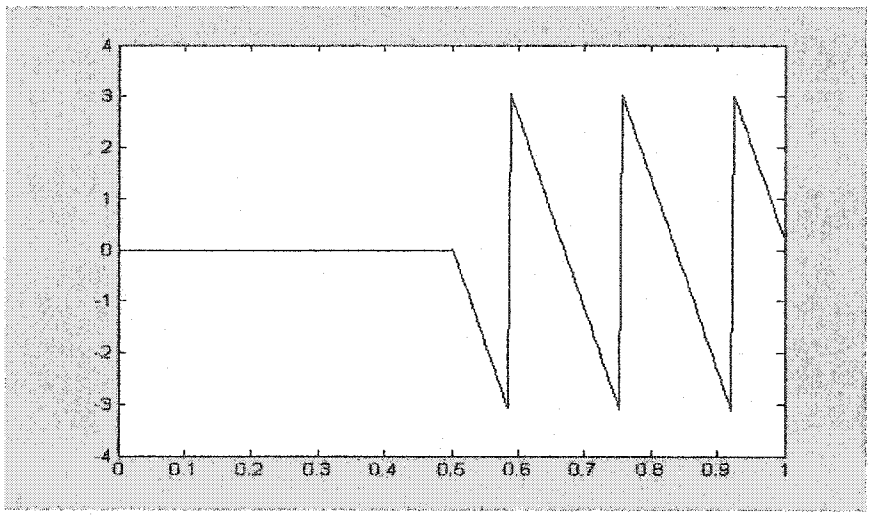


Figure 1.31: Phase of the desired filter (Example 1.6)

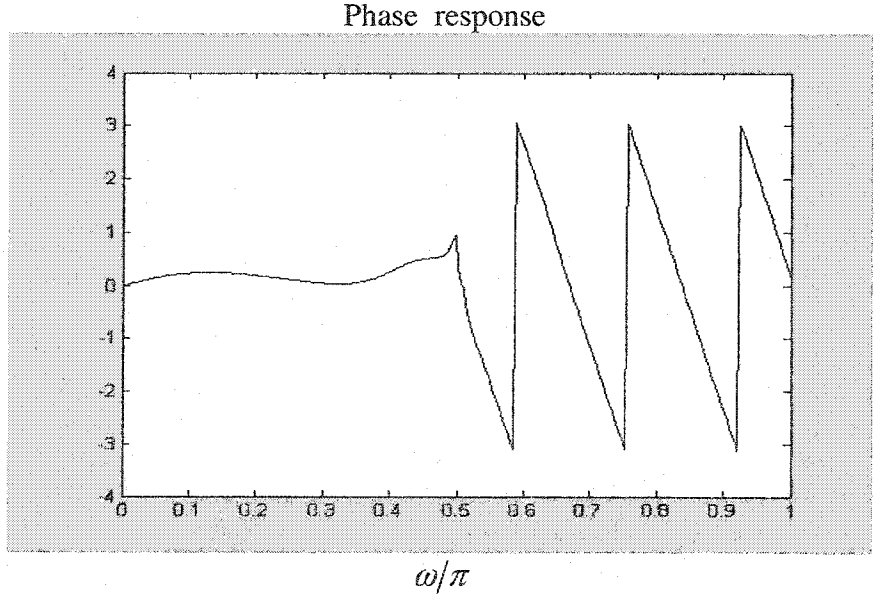


Figure 1.32: Phase of the proposed filter (max. pole radius =0.9913, Example 1.6)

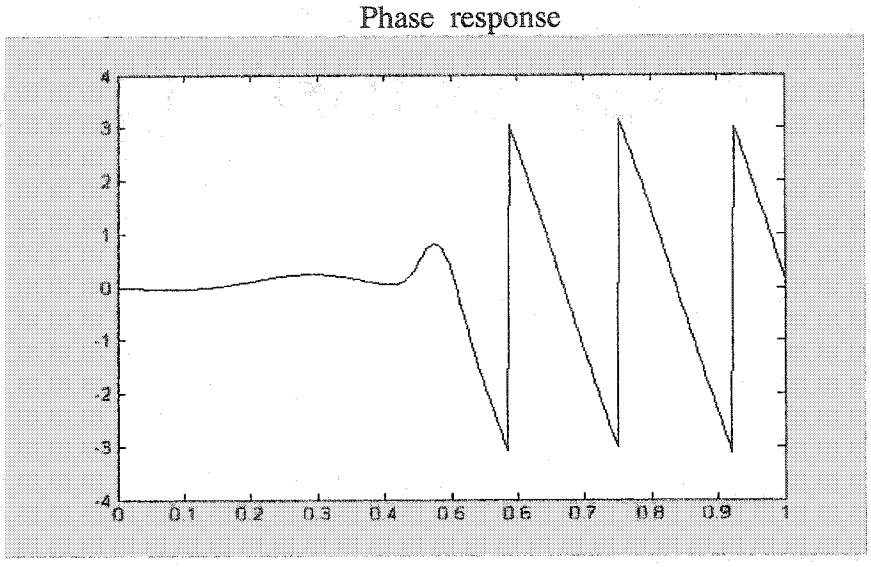


Figure 1.33: Phase of the proposed filter (max. pole radius =0.9276, Example 1.6)

The comparison results are shown in the following tables.

Table 1.4 Frequency response error comparison results of example 1.6

	Gauss-Newton method	Proposed algorithm 2	Damped L-M method	Paper [Lu, (1998)]	Paper [A.T., (1999)]
$E(x)$	25.0003	1.1645	1.2761	2.3227	16.1266
Max.radius	0.9913	0.9913	0.9913	0.9276	0.9913

Table 1.5 Frequency response error comparison results of example 1.6

	Gauss-Newton method	Proposed algorithm2	Damped L-M method	Paper [Lu, (1998)]	Paper [A.T., (1999)]
$E(x)$	801.1239	1.7776	4.0129	2.3227	16.1266
Max.radius	0.9276	0.9276	0.9276	0.9276	0.9913

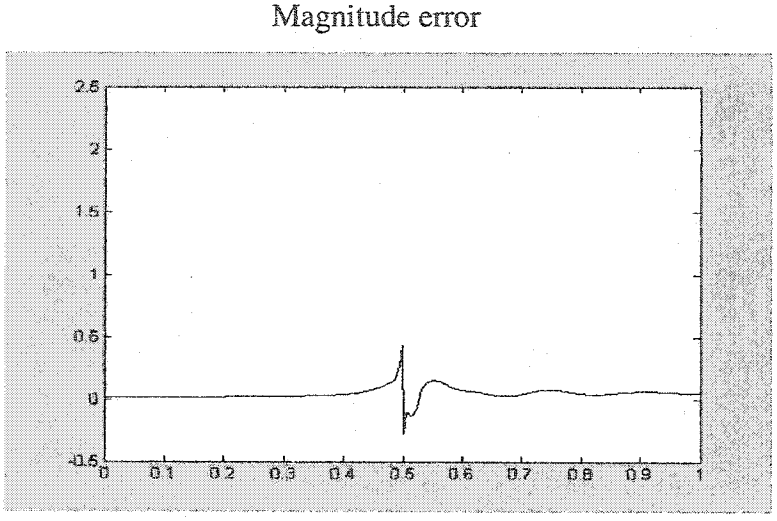


Figure 1.34: Magnitude error of the IIR filter (Gauss-Newton method, table 1.4)

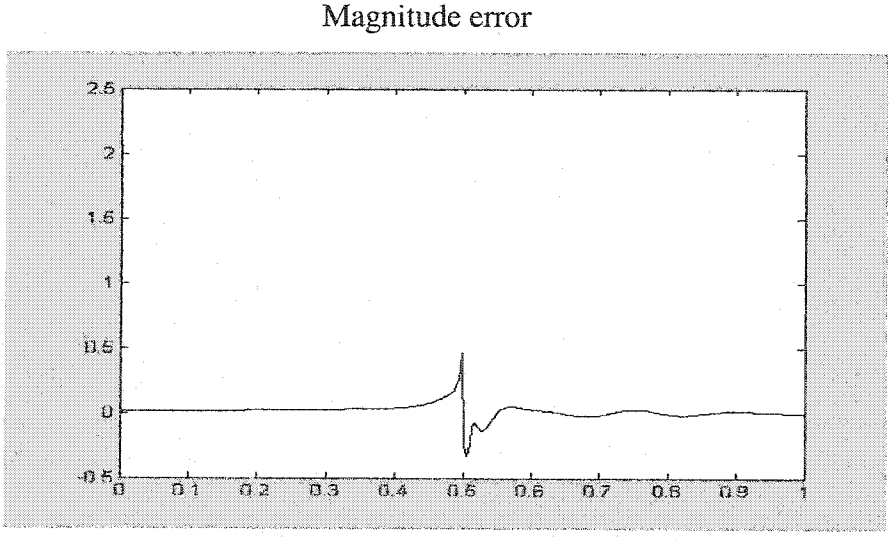


Figure 1.35: Magnitude error of the IIR filter (Proposed algorithm 2, table 1.4)

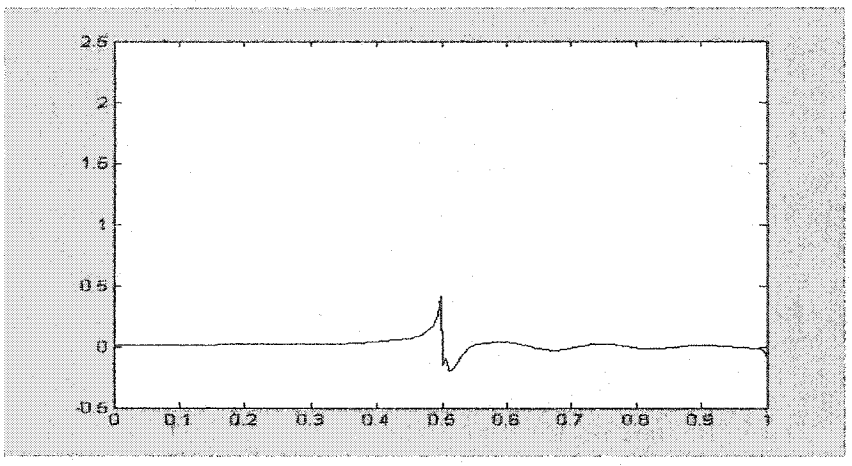


Figure 1.36: Magnitude error of the IIR filter (damped Levenberg-Marquardt method , table 1.4)

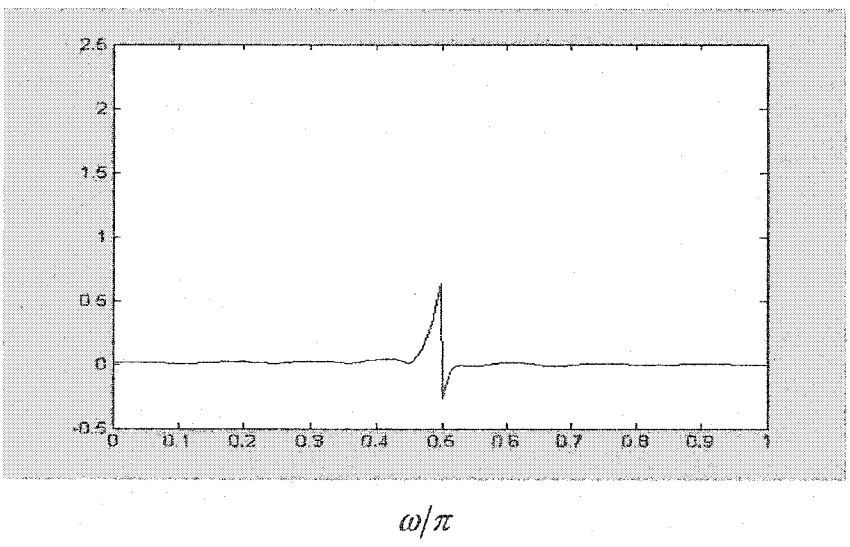


Figure 1.37: Magnitude error of the IIR filter (paper [Lu,(1998)], table 1.4)

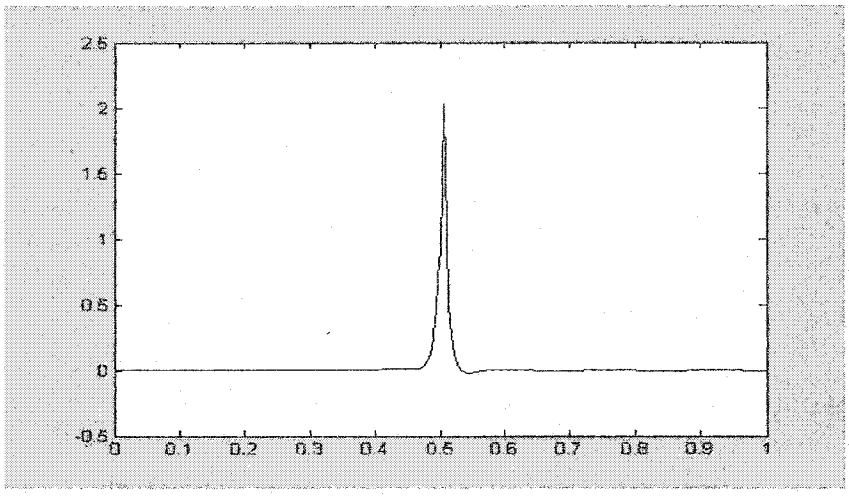


Figure 1.38: Magnitude error of the IIR filter (paper [A.T.,(1999)], table 1.4)

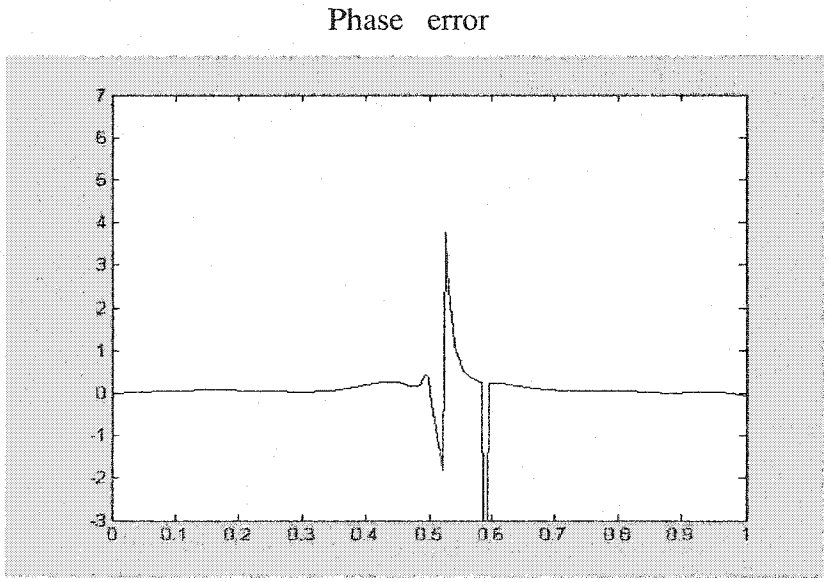


Figure 1.39: Phase error of the IIR filter (Gauss-Newton method , table 1.4)

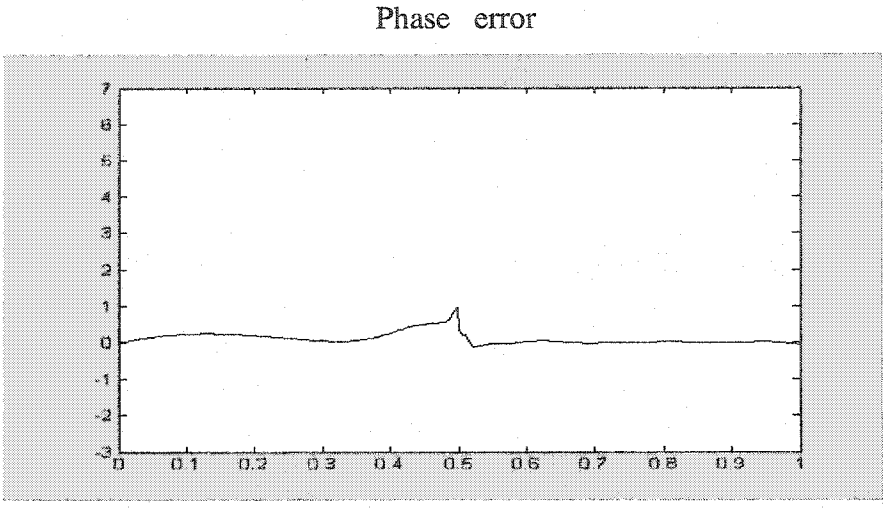


Figure 1.40: Phase error of the IIR filter (Proposed algorithm 2, table 1.4)

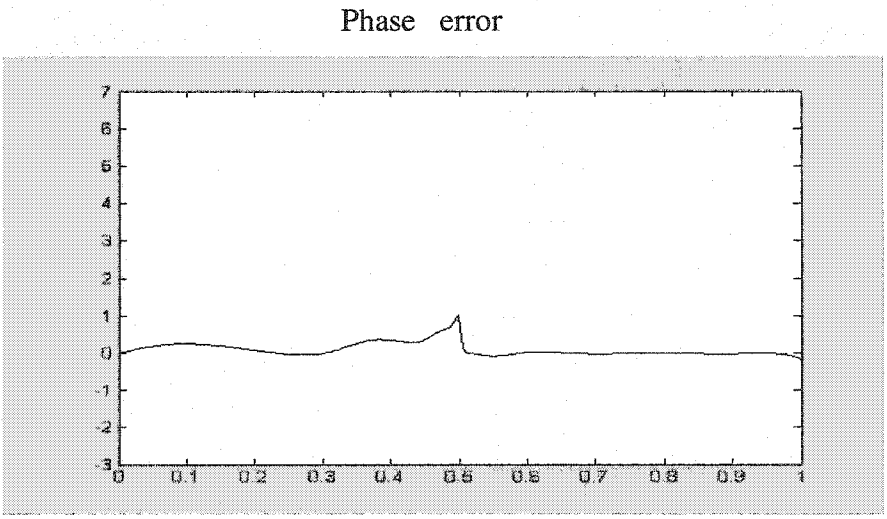
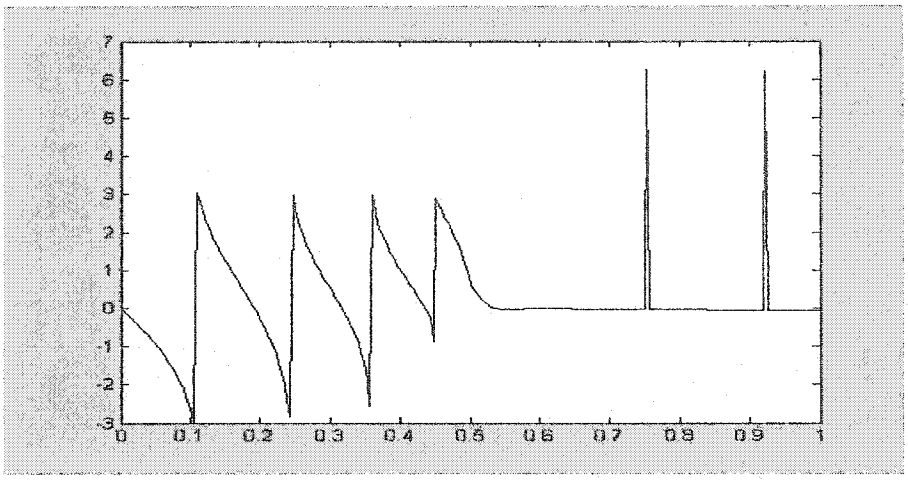


Figure 1.41: Phase error of the IIR filter (damped Levenberg-Marquardt method , table 1.4)

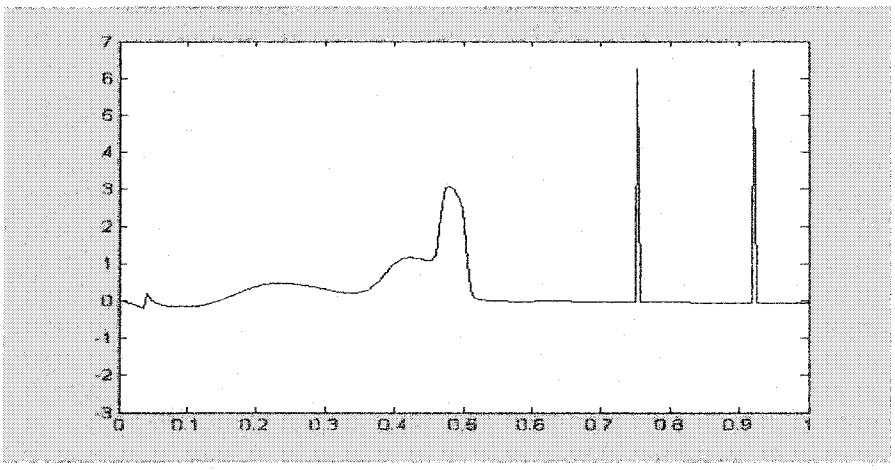
Phase error



ω/π

Figure 1.42: Phase error of the IIR filter (paper [Lu,(1998)], table 1.4)

Phase error



ω/π

Figure 1.43: Phase error of the IIR filter (paper [A.T.,(1999)], table 1.4)

- Example 1.7

The desired frequency [Lu, (1998), A. Tarczynski, (1999)] is the same as example 1.6, it is given by:

$$H_d(e^{j\omega}) = \begin{cases} e^{-12j\omega} & \omega \geq 0.5\pi \\ 0 & \omega < 0.5\pi \end{cases}$$

The order of this filter in [Lu,(1998)] and [A.Tarczynski (1999)] is 14. In this example, the proposed filter order is chosen as $M = N = 10$. Let the proposed filter frequency sampled points $L = 256$ between $0 \sim \pi$. The coefficients of this filter have been published in [Lu,(1998)]. The maximum pole radius of this filter in [Lu,(1998)] is 0.9276. The coefficients of this filter also have been published in [A. Tarczynski, (1999)]. The maximum pole radius of this filter in [A. Tarczynski, (1999)] is 0.9913. In order to make fair comparison, two comparisons are made. Firstly, the maximum radius of the poles is set as 0.9913. Secondly, the maximum radius of the poles is set as 0.9276.

By using *Proposed algorithm 2*, the proposed filters are designed. When the proposed filter maximum poles radius is 9.9130×10^{-1}, the proposed filter coefficients are:

Numerator coefficients: $b = -8.8613\times10^{-5}, 3.1051\times10^{-2}, 1.1563\times10^{-1}, 2.2229\times10^{-1}, 2.8288\times10^{-1}, 2.8384\times10^{-1}, 2.6983\times10^{-1}, 2.5976\times10^{-1}, 2.0793\times10^{-1}, 1.3824\times10^{-1}, 1.2821\times10^{-1}$.

Denominator coefficients: $a = 1.0000, 3.8030, 8.4151, 1.3268\times10^{1}, 1.5968\times10^{1}, 1.5079\times10^{1}, 1.1216\times10^{1}, 6.4705, 2.7737, 8.0307\times10^{-1}, 1.2034\times10^{-1}$.

The frequency response error of the proposed filter is:

$$E(x) = 1.2969.$$

When the proposed filter maximum poles radius is 9.2760×10^{-1}, the proposed filter coefficients are:

Numerator coefficients: $b = -3.8708\times10^{-5}, 3.0989\times10^{-2}, 1.1250\times10^{-1}, 2.0131\times10^{-1}, 2.1954\times10^{-1}, 1.6481\times10^{-1}, 1.0904\times10^{-1}, 8.6246\times10^{-2}, 4.8910\times10^{-2}, 2.1368\times10^{-2}, 7.7755\times10^{-2}.$

Denominator coefficients: $a = 1.0000, 3.7624, 8.0532, 1.2146\times10^{1}, 1.3864\times10^{1}, 1.2306\times10^{1}, 8.5139, 4.5060, 1.7386, 4.4016\times10^{-1}, 5.4639\times10^{-2}.$

The frequency response error of the proposed filter is:

$$E(x) = 1.4674.$$

1.4.3 Proposed algorithm 3

Proposed algorithm 3 is the modification of Lang algorithm [M.C. Lang,(2000)]. Lang designed IIR filters with arbitrary frequency response and a pole radius constraint. Lang used Gauss-Newton method, quadratic programming and the multiple exchange algorithm.

Considering equation (1.12), a coefficients vector is defined as:

$$x = [b_0, b_1, b_2, b_3, \ldots\ldots, b_M, a_1, a_2, a_3, \ldots\ldots, a_N]^T$$

where coefficients a,b are real values. Now, define a vector δ.

Let $$\delta_{(k)} = x_{k+1} - x_k \tag{1.36}$$

According to paper [M.C.Lang,(2000)], minimizing equation (1.13) can be written as

$$\text{Minimize} \quad \frac{1}{2}\delta_{(k)}{}^{T} H_{(k)} \delta_{(k)} - \nabla E_{(k)}{}^{T} \delta_{(k)} \tag{1.37}$$

where $H_{(k)} = 2J_{(k)}{}^{H} J_{(k)}$, $\nabla E_{(k)} = \mathrm{Re}\left\{2J_{(k)}{}^{H} f_{(k)}\right\}$.

Solving the minimization problem equation (1.37) is equivalent to computing the solution to the system of linear equations,

$$H_{(k)}\delta_{(k)} = -\nabla E_{(k)}$$

This is the Gauss-Newton method. Then the coefficients vector can be obtained in the k_{th} iteration. In order to apply damped Gauss-Newton method, a positive scalar α_k is introduced at the k_{th} iteration. From equation (1.36), the new update value of x can be obtained.

$$x_{k+1} = x_k + \alpha_k \delta_{(k)} \qquad (0 < \alpha_k \leq 1) \tag{1.38}$$

Define $\Delta(z) = \delta_{M+2} z^{-1} + \delta_{M+3} z^{-2} + \ldots\ldots + \delta_{M+N+1} z^{-N}$,

where $\delta_n (n = M+2, M+3, ..., M+N+1)$, are the last N elements of the update vector $\delta_{(k)}$, i.e., δ_n is the update values of the denominator coefficients a. According to equation (1.38), the new denominator polynomial is given by:

$$A_{(k+1)}(z) = A_{(k)}(z) + \alpha_k \Delta_{(k)}(z) \qquad 0 < \alpha_k \leq 1$$

where $\Delta_{(k)}(z)$ is the denominator update in the k_{th} iteration. If an initial polynomial $A_{(1)}(z)$ is chosen with all its zeros inside a circle of radius ρ, and if the updates $\Delta_{(k)}(z)$, $k = 1,2,3,.....$ satisfy the following inequality,

$$\left|\Delta_{(k)}(z)\right| \leq \left|A_{(k)}(z)\right| \qquad |z| = \rho \qquad (1.39)$$

then all polynomials $A_{(k)}(z)$, $k = 1,2,3,.....$ have their zeros inside this circle [M.C. Lang,(2000)]. In every iteration, an optimal update $\delta_{(k)}$ is computed for satisfying the requirement of the maximum pole radius. Let δ is partitioned in vectors δ_b and δ_a. Vector δ_b contains the first $M+1$ elements updating the numerator coefficients b_m, $m = 0,1,2,.....,M$. Vector δ_a contains the last N elements of δ that update the denominator coefficients a_n, $n = 1,2,3,.....,N$.

$$\delta = \begin{bmatrix} \delta_b \\ \\ \delta_a \end{bmatrix}$$

According to equation (1.37) and (1.39), in order to obtain stable IIR filter with the maximum pole radius ρ, the following quadratic programming problem must to be solved to obtain the update $\delta_{(k)}$ at the k_{th} iteration [M.C. Lang,(2000)].

$$\text{minimize} \qquad \frac{1}{2}\delta_{(k)}{}^{T} H_{(k)} \delta_{(k)} - \nabla E_{(k)}{}^{T} \delta_{(k)}$$

$$\text{subject to} \qquad \left|\Delta_{(k)}(\rho e^{j\theta})\right| \le \left|A_{(k)}(\rho e^{j\theta})\right| \qquad \theta \in [0,\pi] \tag{1.40}$$

After obtaining $\delta_{(k)}$, according to equation (1.38), damped Gauss-Newton method can be used to obtain x_{k+1}. Now, we will show the equivalent inequality of

$$\left|\Delta_{(k)}(\rho e^{j\theta})\right| \le \left|A_{(k)}(\rho e^{j\theta})\right| \tag{1.41}$$

$$\text{where } \left|\Delta_{(k)}(\rho e^{j\theta})\right| \approx \mathrm{Re}\left\{\Delta_{(k)}(\rho e^{j\theta}) \cdot e^{-j\phi\Delta_{(k)}(\theta)}\right\} \approx \mathrm{Re}\left\{\Delta_{(k)}(\rho e^{j\theta}) \cdot e^{-j\phi\Delta_{(k-1)}(\theta)}\right\} \tag{1.42}$$

where $\phi\Delta(\theta)$ is defined by $\phi\Delta(\theta) = \arg\left\{\Delta(\rho e^{j\theta})\right\}$.

According to equation (1.42), equation (1.41) can be rewritten in the matrix form. Hence,

$$\left|\Delta_{(k)}(\rho e^{j\theta})\right| \approx \mathrm{Re}\left\{\Delta_{(k)}(\rho e^{j\theta}) \cdot e^{-j\phi\Delta_{(k-1)}(\theta)}\right\}$$

$$= \begin{pmatrix} \rho^{-1}\cos(j\omega_1 + \phi\Delta_{(k-1)}(\theta)) & \cdots\cdots & \rho^{-N}\cos(Nj\omega_1 + \phi\Delta_{(k-1)}(\theta)) \\ \rho^{-1}\cos(j\omega_2 + \phi\Delta_{(k-1)}(\theta) & \cdots\cdots & \rho^{-N}\cos(Nj\omega_2 + \phi\Delta_{(k-1)}(\theta)) \\ \vdots & \cdots\cdots & \vdots \\ \vdots & \cdots\cdots & \vdots \\ \rho^{-1}\cos(j\omega_L + \phi\Delta_{(k-1)}(\theta)) & \cdots\cdots & \rho^{-N}\cos(Nj\omega_L + \phi\Delta_{(k-1)}(\theta)) \end{pmatrix} \begin{pmatrix} \delta_{M+1} \\ \delta_{M+2} \\ \vdots \\ \vdots \\ \delta_{M+N} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & \cdots & 0 & \rho^{-1}\cos(j\omega_1 + \phi\Delta_{(k-1)}(\theta)) & \cdots & \rho^{-N}\cos(Nj\omega_1 + \phi\Delta_{(k-1)}(\theta)) \\ \vdots & \cdots & \vdots & \rho^{-1}\cos(j\omega_2 + \phi\Delta_{(k-1)}(\theta)) & \cdots & \rho^{-N}\cos(Nj\omega_2 + \phi\Delta_{(k-1)}(\theta)) \\ \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \rho^{-1}\cos(j\omega_L + \phi\Delta_{(k-1)}(\theta)) & \cdots & \rho^{-N}\cos(Nj\omega_L + \phi\Delta_{(k-1)}(\theta)) \end{pmatrix} \begin{pmatrix} \delta_0 \\ \delta_1 \\ \vdots \\ \delta_{M+N} \end{pmatrix}$$

$$= C_{(k)}\delta_{(k)}$$

where matrix $C_{(k)}$ is a $L \times (M+N+1)$ matrix, $\delta_{(k)}$ is a $(M+N+1) \times 1$ vector.

$$\left|A_{(k)}(\rho e^{j\theta})\right| =$$

$$abs\left\{\begin{pmatrix} 1 & \rho^{-1}e^{-j\omega_1} & \rho^{-2}e^{-2j\omega_1} & \dots & \rho^{-N}e^{-Nj\omega_1} \\ 1 & \rho^{-1}e^{-j\omega_2} & \rho^{-2}e^{-2j\omega_2} & \dots & \rho^{-N}e^{-Nj\omega_2} \\ 1 & \rho^{-1}e^{-j\omega_3} & \rho^{-2}e^{-2j\omega_3} & \dots & \rho^{-N}e^{-Nj\omega_3} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \rho^{-1}e^{-j\omega_L} & \rho^{-2}e^{-2j\omega_L} & \dots & \rho^{-N}e^{-Nj\omega_L} \end{pmatrix} \begin{pmatrix} 1 \\ x_{M+2} \\ x_{M+3} \\ \vdots \\ \vdots \\ x_{M+N+1} \end{pmatrix}\right\}$$

Therefore, Equation (1.40) can be written as the standard quadratic programming problem:

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2}\delta_{(k)}{}^{T} H_{(k)} \delta_k - \nabla E_{(k)}{}^{T} \delta_{(k)} \\ &\text{Subject to} \quad C_{(k)} \cdot \delta_{(k)} \le \left|A_{(k)}(\rho e^{j\theta})\right| \end{aligned} \tag{1.43}$$

In [M.C.Lang,(2000)], Lang used a multiple exchange algorithm to determine the angle $\phi\Delta_{(k)}(\theta)$ in (1.42). In our algorithm, the angle $\phi\Delta_{(k-1)}(\theta)$ of the $(k-1)_{th}$ iteration is used to replace the angle $\phi\Delta_{(k)}(\theta)$ of the k_{th} iteration in equation (1.42). Through such modification, the algorithm is simplified. Several design examples are tested. The test results show that our algorithm can also obtain good results. The other difference between Lang algorithm and our algorithm is that Gauss-Newton method is used in Lang algorithm, while damped Gauss-Newton method is used in our algorithm.

Proposed algorithm 3 works as follows:

a) Choose an initial stable guess x_1 with the prescribed stable region. (i.e., maximum pole radius is smaller than or equal to ρ), section 1.1 least squares linear equation error IIR filters design algorithm is used to choose x_1.

   Set $k = 1, tol = 0.01$.

b) Compute $E(x_k)$ from (1.14).

Compute $\nabla E_{(k)}$ from (1.27).

Compute $H_{(k)}$ from (1.28).

c) Solve quadratic programming problem (1.43) to obtain $\delta_{(k)}$.

d) Set $\alpha_k = 1$, solve x_{k+1} from (1.38).

e) Compute $E(x_{k+1})$ from (1.14).

f) While $E(x_{k+1}) > E(x_k)$

$\alpha_k = \alpha_k / 2$.

Compute x_{k+1} again from (1.38).

Compute $E(x_{k+1})$ from (1.14).

endwhile.

g) if $\left\| \frac{x_{k+1} - x_k}{\alpha_k} \right\|_2 < tol$

Stop.

else set $k = k + 1$,

Go back to step (b).

Begin another iteration.

endif

1.4.4 Examples:

Several examples are used to compare Gauss-Newton method, *Proposed algorithm 2*, damped Levenberg-Marquardt method, *Proposed algorithm 3*.

- Example 1.8:

The desired filter is a fourth order lowpass elliptic filter with 0.5 decibel of peak-to-peak ripple and a minimum stopband attenuation of 20 decibels. The normalized

cutoff frequency is 0.6. From Matlab function [b,a]=ellip(4,0.5,20,0.6), the desired filter coefficients b, a can be obtained.

The desired filter coefficients are:

$$b = 3.2091\times10^{-1},\ 8.0481\times10^{-1},\ 1.1046,\ 8.0481\times10^{-1},\ 3.2091\times10^{-1}.$$
$$a = 1.0000,\ 7.8730\times10^{-1},\ 1.2361,\ 3.0624\times10^{-1},\ 2.2524\times10^{-1}.$$

From Matlab function [$H_d(\omega)$]=freqz(b,a,10), the desired filter frequency response can be obtained. The samples of the desired filter frequency response $H_d(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$H_d(\omega) = 9.4406\times10^{-1},$$
$$9.3360\times10^{-1} - 1.7260\times10^{-1}\,i,$$
$$8.9359\times10^{-1} - 3.6453\times10^{-1}\,i,$$
$$7.8977\times10^{-1} - 5.9278\times10^{-1}\,i,$$
$$5.3789\times10^{-1} - 8.4263\times10^{-1}\,i,$$
$$2.1656\times10^{-2} - 9.6139\times10^{-1}\,i,$$
$$-9.4381\times10^{-1} - 2.1546\times10^{-2}\,i,$$
$$-5.1305\times10^{-2} - 8.3081\times10^{-2}\,i,$$
$$6.6294\times10^{-4} + 4.1237\times10^{-4}\,i,$$
$$7.3391\times10^{-2} + 1.8961\times10^{-2}\,i.$$

The samples of the desired filter frequency response magnitude $|H_d(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$|H_d(\omega)| = 9.4406\times10^{-1},\ 9.4942\times10^{-1},\ 9.6508\times10^{-1},\ 9.8749\times10^{-1},$$
$$9.9967\times10^{-1},\ 9.6164\times10^{-1},\ 9.4406\times10^{-1},\ 9.7645\times10^{-2},$$
$$7.8073\times10^{-4},\ 7.5801\times10^{-2}.$$

By using *Proposed algorithm 3*, the proposed filter is designed. The proposed filter maximum pole radius is set as 9.3500×10^{-1}.

The proposed filter coefficients are:

$$b = 3.2091\times10^{-1},\ 8.0481\times10^{-1},\ 1.1046,\ 8.0481\times10^{-1},\ 3.2091\times10^{-1}.$$

$$a = 1.0000,\ 7.8715\times10^{-1},\ 1.2356,\ 3.0613\times10^{-1},\ 2.2513\times10^{-1}.$$

The samples of the proposed filter frequency response $H(\omega)$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$H(\omega) = 9.4428\times10^{-1},$$
$$9.3380\times10^{-1} - 1.7269\times10^{-1}\,i,$$
$$8.9374\times10^{-1} - 3.6472\times10^{-1}\,i,$$
$$7.8982\times10^{-1} - 5.9308\times10^{-1}\,i,$$
$$5.3770\times10^{-1} - 8.4299\times10^{-1}\,i,$$
$$2.0983\times10^{-2} - 9.6151\times10^{-1}\,i,$$
$$-9.4104\times10^{-1} - 2.0387\times10^{-2}\,i,$$
$$-5.1248\times10^{-2} - 8.3122\times10^{-2}\,i,$$
$$6.6295\times10^{-4} + 4.1264\times10^{-4}\,i,$$
$$7.3405\times10^{-2} + 1.8973\times10^{-2}\,i.$$

The samples of the proposed filter frequency response magnitude $|H(\omega)|$ at a frequency spacing of $\frac{\pi}{9}$ are given by:

$$|H(\omega)| = 9.4428\times10^{-1},\ 9.4964\times10^{-1},\ 9.6530\times10^{-1},\ 9.8771\times10^{-1},$$
$$9.9987\times10^{-1},\ 9.6173\times10^{-1},\ 9.4126\times10^{-1},\ 9.7651\times10^{-2},$$
$$7.8088\times10^{-4},\ 7.5817\times10^{-2}.$$

The proposed filter poles radius are:

$$9.3500\times10^{-1}, 9.3500\times10^{-1}, 5.0746\times10^{-1}, 5.0746\times10^{-1}.$$

The proposed filter frequency response error:

$$E(x)=9.9558\times10^{-6}.$$

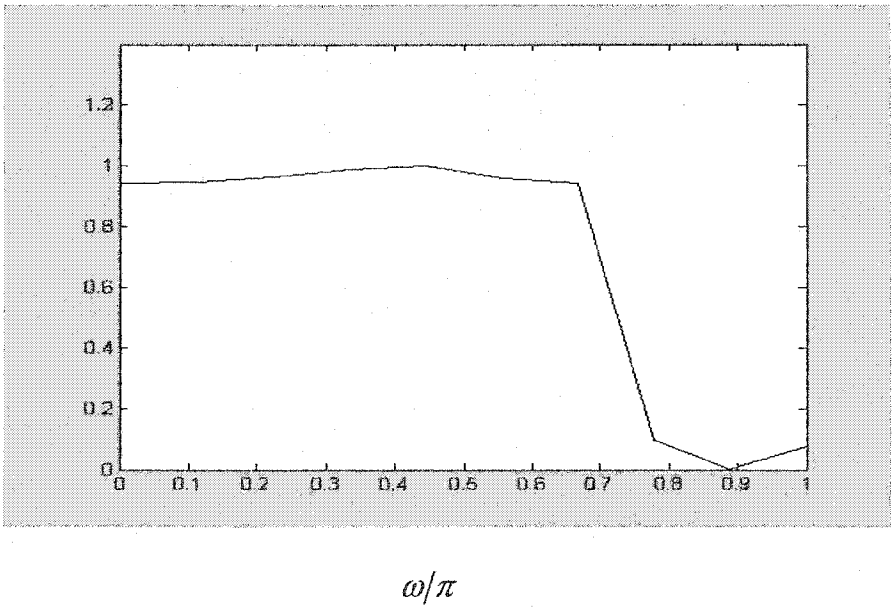


Figure 1.44: Magnitude of the desired filter (Example 1.8)

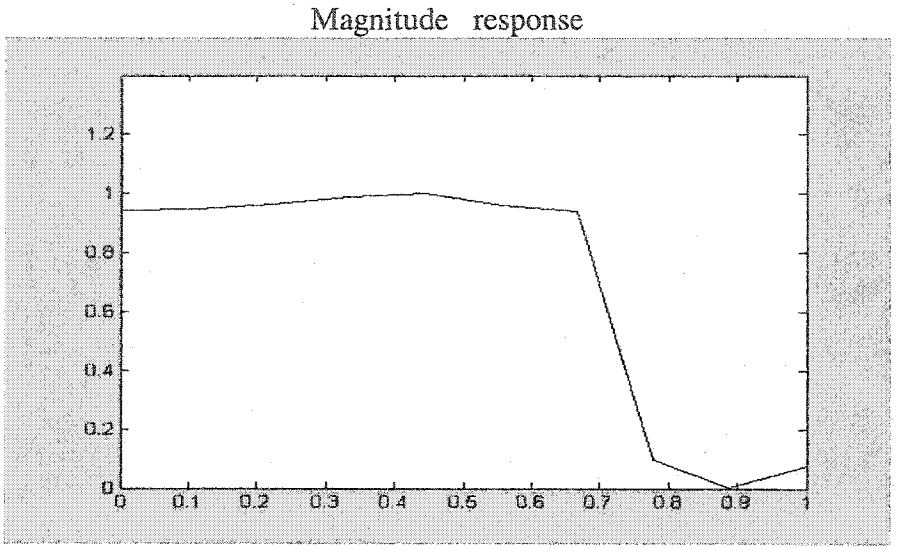


Figure 1.45: Magnitude of the proposed filter (Example 1.8)

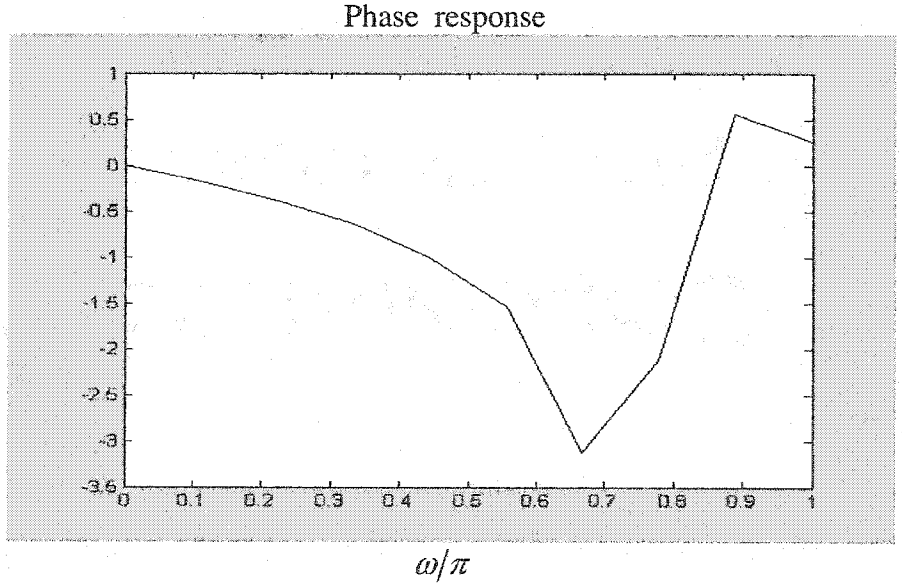


Figure 1.46: Phase of the desired filter (Example 1.8)

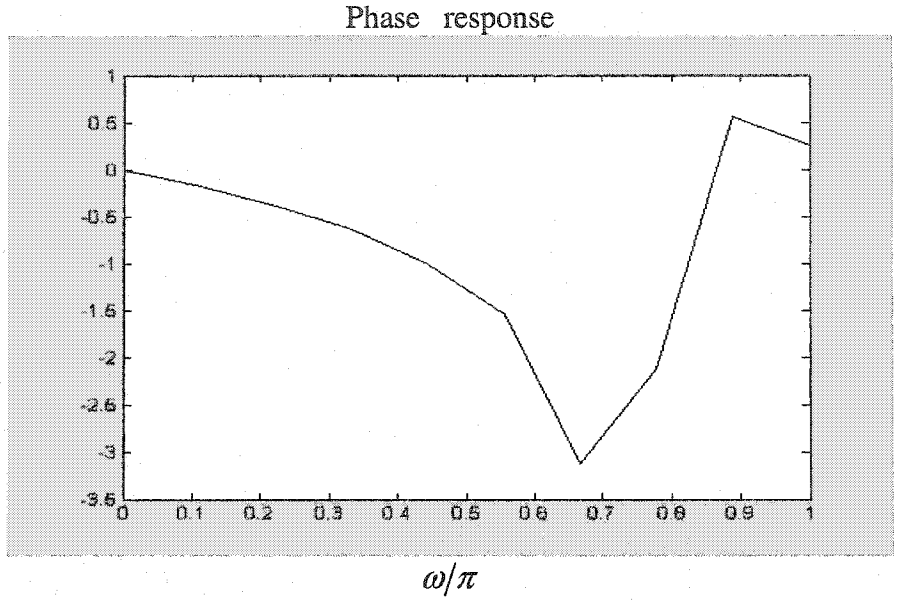


Figure 1.47: Phase of the proposed filter (Example 1.8)

- Example 1.9

This example was originally reported by Lu in [W.S. Lu,(1997)]. Lu designed a 15 order low-pass filter. The coefficients has been published in [W.S.Lu,(1997)]. The frequency response of this filter is used as the desired frequency response. Let the proposed filter order $M = N = 15$. Let the proposed filter frequency sampled points $L = 100$ between $0 \sim \pi$.

The coefficients of the desired filter is shown in table 1.6.

Table 1.6 Desired filter coefficients of example 1.9 ([W.S.Lu,(1997)])

Denominator coefficients	Numerator coefficients
1.00000e+0	-5.22654e-3
-1.56463e+0	1.94457e-2
1.73980e+0	-1.15035e-2

-1.57008e+0	-7.20118e-3
1.39721e+0	5.93816e-5
-1.27637e+0	2.79188e-2
1.10667e+0	-1.48195e-2
-9.11686e-1	-2.49149e-2
7.59812e-1	2.88456e-3
-6.20832e-1	5.06928e-2
4.59430e-1	-1.14557e-2
-3.16294e-1	-6.05705e-2
2.19661e-1	-2.08870e-2
-1.39322e-1	1.26918e-1
6.19802e-2	1.60440e-1
-1.31287e-2	1.03666e-1

The maximum pole radius of this filter is 0.8263. In order to show the proposed algorithm can be used to design stable IIR filters with specified stable region,(i.e., the maximum pole radius can be set arbitrarily), three comparisons are made. Firstly, the maximum radius of the poles is set as 0.8263. Secondly, the maximum radius of the poles is set as 0.8260. Thirdly, the maximum radius of the poles is set as 0.8250.

By using *Proposed algorithm 3*, the proposed filters are designed.
When the proposed filter maximum pole radius is 8.2630×10^{-1}, the proposed filter coefficients are:

$$b = -5.2265\times10^{-3}, 1.9446\times10^{-2}, -1.1503\times10^{-2}, -7.2012\times10^{-3},$$
$$5.9382\times10^{-5}, 2.7919\times10^{-2}, -1.4819\times10^{-2}, -2.4915\times10^{-2},$$
$$2.8846\times10^{-3}, 5.0693\times10^{-2}, -1.1456\times10^{-2}, -6.0571\times10^{-2},$$
$$-2.0887\times10^{-2}, 1.2692\times10^{-1}, 1.6044\times10^{-1}, 1.0367\times10^{-1}.$$

a = 1.0000, -1.5646, 1.7398, -1.5701, 1.3972, -1.2764, 1.1067,
-9.1169×10^{-1}, 7.5981×10^{-1}, -6.2083×10^{-1}, 4.5943×10^{-1},
-3.1629×10^{-1}, 2.1966×10^{-1}, -1.3932×10^{-1}, 6.1980×10^{-2},
-1.3129×10^{-2}.

The proposed filter poles radius:

8.2259×10^{-1}, 8.2259×10^{-1}, 7.8321×10^{-1}, 7.8321×10^{-1},
7.9440×10^{-1}, 7.9440×10^{-1}, 8.2629×10^{-1}, 8.2629×10^{-1}, 7.3829×10^{-1}, 7.3829×10^{-1},
6.9381×10^{-1}, 6.9381×10^{-1}, 6.6108×10^{-1}, 6.6108×10^{-1}, 6.4020×10^{-1}.

When the proposed filter maximum pole radius is 8.2600×10^{-1}, the proposed filter coefficients are:

b = -5.2265×10^{-3}, 1.9446×10^{-2}, -1.1503×10^{-2}, -7.2012×10^{-3},
5.9382×10^{-5}, 2.7919×10^{-2}, -1.4819×10^{-2}, -2.4915×10^{-2},
2.8846×10^{-3}, 5.0693×10^{-2}, -1.1456×10^{-2}, -6.0571×10^{-2},
-2.0887×10^{-2}, 1.2692×10^{-1}, 1.6044×10^{-1}, 1.0367×10^{-1}.

a = 1.0000, -1.5645, 1.7392, -1.5695, 1.3969, -1.2760, 1.1062,
-9.1137×10^{-1}, 7.5959×10^{-1}, -6.2059×10^{-1}, 4.5923×10^{-1},
-3.1618×10^{-1}, 2.1958×10^{-1}, -1.3925×10^{-1}, 6.1940×10^{-2},
-1.3120×10^{-2}.

The proposed filter poles radius:

8.2259×10^{-1}, 8.2259×10^{-1}, 7.8321×10^{-1}, 7.8321×10^{-1}, 7.9440×10^{-1}, 7.9440×10^{-1}, 8.2600×10^{-1}, 8.2600×10^{-1}, 7.3829×10^{-1}, 7.3829×10^{-1}, 6.9381×10^{-1}, 6.9381×10^{-1}, 6.6108×10^{-1}, 6.6108×10^{-1}, 6.4020×10^{-1}.

When the proposed filter maximum pole radius is 8.2500×10^{-1}, the proposed filter coefficients are:

$$b = 5.2265\times10^{-3}, 1.9446\times10^{-2}, -1.1503\times10^{-2}, -7.2012\times10^{-3}, 5.9382\times10^{-5}, 2.7919\times10^{-2}, -1.4819\times10^{-2}, -2.4915\times10^{-2}, 2.8846\times10^{-3}, 5.0693\times10^{-2}, -1.1456\times10^{-2}, -6.0571\times10^{-2}, -2.0887\times10^{-2}, 1.2692\times10^{-1}, 1.6044\times10^{-1}, 1.0367\times10^{-1}.$$

$$a = 1.0000, -1.5639, 1.7369, -1.5673, 1.3956, -1.2747, 1.1047, -9.1028\times10^{-1}, 7.5883\times10^{-1}, -6.1977\times10^{-1}, 4.5853\times10^{-1}, -3.1579\times10^{-1}, 2.1931\times10^{-1}, -1.3900\times10^{-1}, 6.1801\times10^{-2}, -1.3088\times10^{-2}.$$

The proposed filter poles radius:

8.2259×10^{-1}, 8.2259×10^{-1}, 7.8321×10^{-1}, 7.8321×10^{-1}, 7.9440×10^{-1}, 7.9440×10^{-1}, 8.2500×10^{-1}, 8.2500×10^{-1}, 7.3829×10^{-1}, 7.3829×10^{-1}, 6.9381×10^{-1}, 6.9381×10^{-1}, 6.6108×10^{-1}, 6.6108×10^{-1}, 6.4020×10^{-1}.

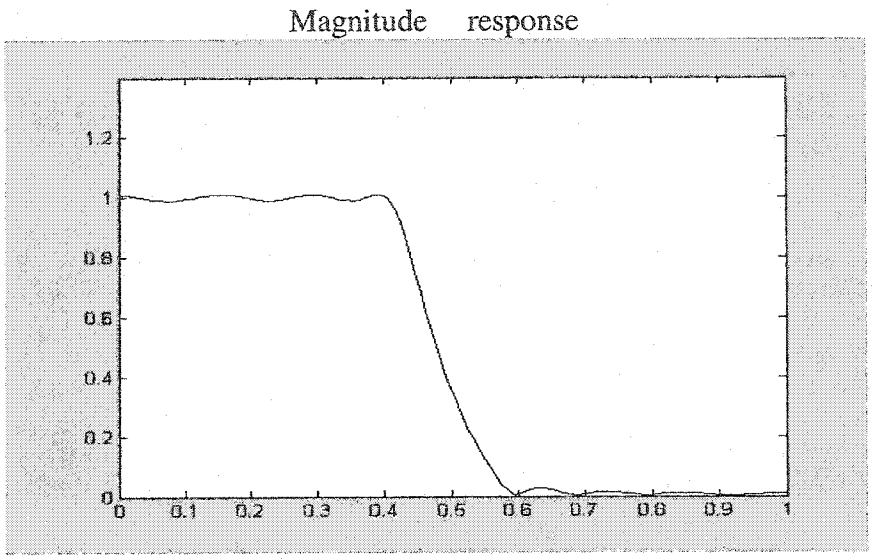


Figure 1.48: Magnitude of the desired filter (Example 1.9)

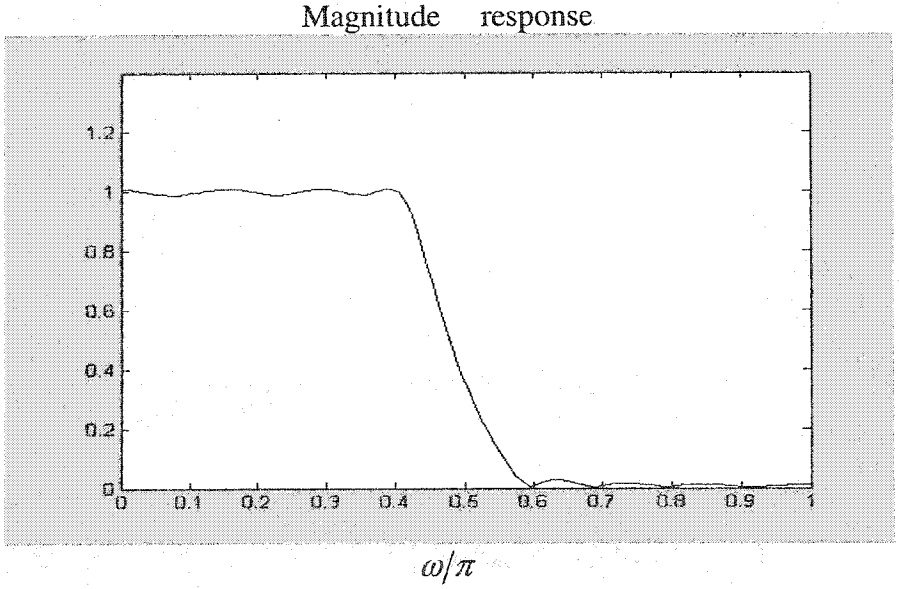


Figure 1.49: Magnitude of the proposed filter (max. pole radius =0.8263, Example 1.9)

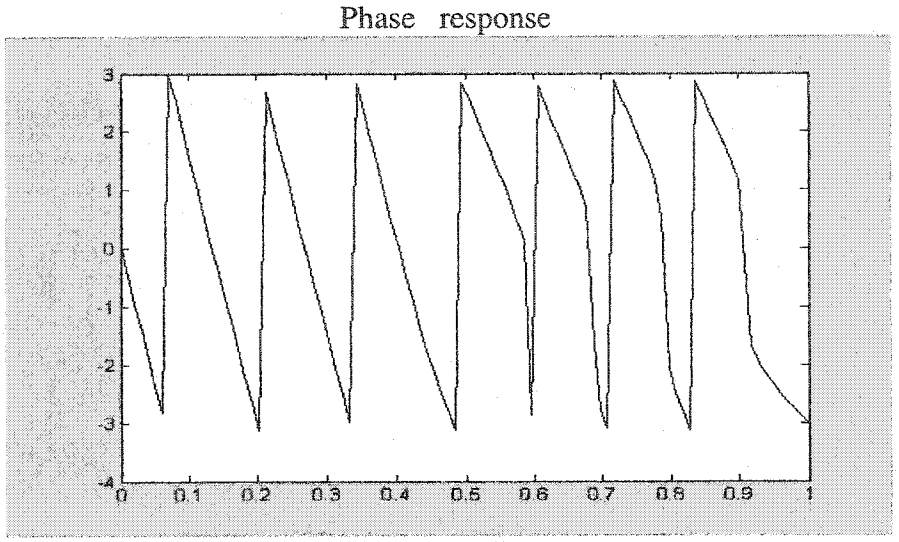


Figure 1.50: Phase of the desired filter (Example 1.9)

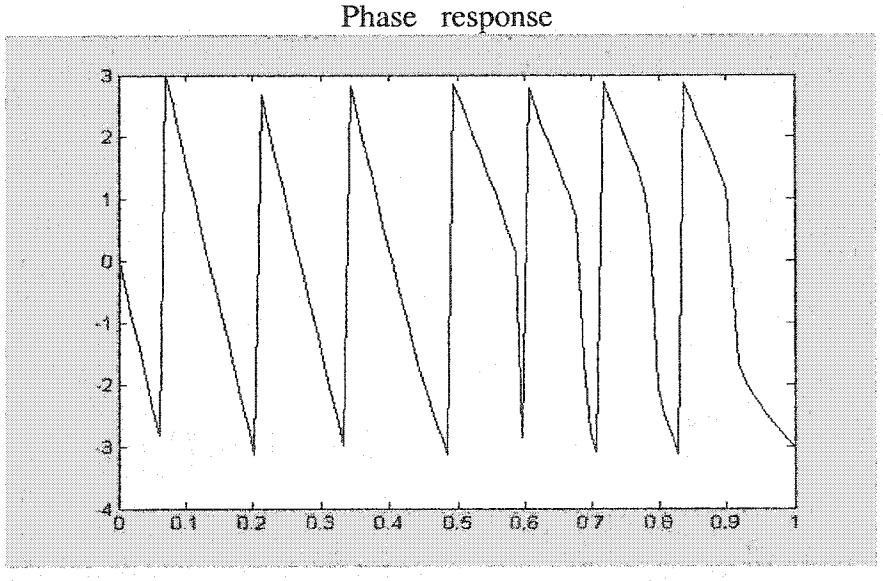


Figure 1.51: Phase of the proposed filter (max. pole radius =0.8263, Example 1.9)

The comparison results are shown in the follwing table:

Table 1.7 Frequency response error of example 1.9

	Gauss-Newton method	Proposed algorithm 2	Damped L-M method	Proposed algorithm 3
$E(x)$ (max.radius: 0.8263)	2.6652 $\times 10^{-27}$	2.5616 $\times 10^{-27}$	2.3073 $\times 10^{-26}$	2.3060 $\times 10^{-25}$
$E(x)$ (max.radius: 0.8260)	2.5782 $\times 10^{-5}$	2.5601 $\times 10^{-5}$	2.5486 $\times 10^{-5}$	2.5601 $\times 10^{-5}$
$E(x)$ (max.radius: 0.8250)	5.2708 $\times 10^{-4}$	5.1080 $\times 10^{-4}$	5.2036 $\times 10^{-4}$	5.1080 $\times 10^{-4}$

- Example 1.10

This example was originally reported by Lu in [Lu,(1998)]. Lu designed a 14 order high-pass filter. The coefficients has been published in [Lu,(1998)]. The maximum pole radius of this filter is 0.9276. The frequency response of this filter is used as the desired frequency response. Three comparisons are made. The maximum pole radius are set as 0.9276, 0.9200, 0.9000, respectively. Let the proposed filter order $M = N = 14$. Let the proposed filter frequency sampled points $L = 100$ between $0 \sim \pi$. The coefficients of the desired filter in [Lu,(1998)] are shown in table 1.8.

Table 1.8 Desired filter coefficients of example 1.10 ([Lu, (1998)])

Denominator coefficients	Numerator coefficients
1.00000000	0.00216452
0.85183505	0.01422263

1.40277206	0.01070862
1.15151797	-0.00173896
0.95948743	-0.00056474
0.81499579	0.01648599
0.68685173	0.01450372
0.52355787	-0.01424925
0.35584783	-0.01101010
0.23561367	0.04991283
0.15595331	0.04637753
0.08260605	-0.19765241
0.01749834	0.33139538
-0.01347975	-0.25569216
-0.01109415	0.13691517

By using *Proposed algorithm 3,* the proposed filters are designed .

When the proposed filter maximum pole radius is 9.2760×10^{-1}, the proposed filter coefficients are:

$$b = 2.1645\times10^{-3},\ 1.4223\times10^{-2},\ 1.0709\times10^{-2},\ -1.7390\times10^{-3},$$
$$-5.6474\times10^{-4},\ 1.6486\times10^{-2},\ 1.4504\times10^{-2},\ -1.4249\times10^{-2},$$
$$-1.1010\times10^{-2},\ 4.9913\times10^{-2},\ 4.6378\times10^{-2},\ -1.9765\times10^{-1},$$
$$3.3140\times10^{-1},\ -2.5569\times10^{-1},\ 1.3692\times10^{-1}.$$

$$a = 1.0000,\ 8.5184\times10^{-1},\ 1.4028,\ 1.1515,\ 9.5949\times10^{-1},$$
$$8.1500\times10^{-1},\ 6.8685\times10^{-1},\ 5.2356\times10^{-1},\ 3.5585\times10^{-1},$$
$$2.3561\times10^{-1},\ 1.5595\times10^{-1},\ 8.2606\times10^{-2},\ 1.7498\times10^{-2},$$

$$-1.3480\times10^{-2}, -1.1094\times10^{-2}.$$

The proposed filter poles radius:

$$9.2757\times10^{-1}, 9.2757\times10^{-1}, 7.5422\times10^{-1}, 7.5422\times10^{-1},$$
$$6.6231\times10^{-1}, 6.6905\times10^{-1}, 6.6905\times10^{-1}, 6.9350\times10^{-1},$$
$$6.9350\times10^{-1}, 7.9592\times10^{-1}, 7.9592\times10^{-1}, 7.9940\times10^{-1},$$
$$7.9940\times10^{-1}, 3.9270\times10^{-1}.$$

When the proposed filter maximum pole radius is 9.2000×10^{-1}, the proposed filter coefficients are:

$$b = 2.1645\times10^{-3}, 1.4223\times10^{-2}, 1.0709\times10^{-2}, -1.7390\times10^{-3},$$
$$-5.6474\times10^{-4}, 1.6486\times10^{-2}, 1.4504\times10^{-2}, -1.4249\times10^{-2},$$
$$-1.1010\times10^{-2}, 4.9913\times10^{-2}, 4.6378\times10^{-2}, -1.9765\times10^{-1},$$
$$3.3140\times10^{-1}, -2.5569\times10^{-1}, 1.3692\times10^{-1}.$$

$$a = 1.0000, 8.5169\times10^{-1}, 1.3887, 1.1398, 9.5205\times10^{-1},$$
$$8.0899\times10^{-1}, 6.7983\times10^{-1}, 5.1735\times10^{-1}, 3.5232\times10^{-1},$$
$$2.3364\times10^{-1}, 1.5401\times10^{-1}, 8.1018\times10^{-2}, 1.7002\times10^{-2},$$
$$-1.3263\times10^{-2}, -1.0914\times10^{-2}.$$

The proposed filter poles radius:

$$9.2000\times10^{-1}, 9.2000\times10^{-1}, 7.5422\times10^{-1}, 7.5422\times10^{-1},$$

$6.6231\times10^{-1}, 6.6905\times10^{-1}, 6.6905\times10^{-1}, 6.9350\times10^{-1},$
$6.9350\times10^{-1}, 7.9592\times10^{-1}, 7.9592\times10^{-1}, 7.9940\times10^{-1},$
$7.9940\times10^{-1}, 3.9270\times10^{-1}.$

When the proposed filter maximum pole radius is 9.0000×10^{-1}, the proposed filter coefficients are:

$$b = 2.1645\times10^{-3}, 1.4223\times10^{-2}, 1.0709\times10^{-2}, -1.7390\times10^{-3},$$
$$-5.6474\times10^{-4}, 1.6486\times10^{-2}, 1.4504\times10^{-2}, -1.4249\times10^{-2},$$
$$-1.1010\times10^{-2}, 4.9913\times10^{-2}, 4.6377\times10^{-2}, -1.9765\times10^{-1},$$
$$3.3140\times10^{-1}, -2.5569\times10^{-1}, 1.3692\times10^{-1}.$$

$$a = 1.0000, 8.5132\times10^{-1}, 1.3520, 1.1092, 9.3267\times10^{-1},$$
$$7.9336\times10^{-1}, 6.6154\times10^{-1}, 5.0119\times10^{-1}, 3.4313\times10^{-1},$$
$$2.2851\times10^{-1}, 1.4896\times10^{-1}, 7.6881\times10^{-2}, 1.5709\times10^{-2},$$
$$-1.2698\times10^{-2}, -1.0445\times10^{-2}.$$

The proposed filter poles radius:

$9.0000\times10^{-1}, 9.0000\times10^{-1}, 7.5422\times10^{-1}, 7.5422\times10^{-1},$
$6.6231\times10^{-1}, 6.6905\times10^{-1}, 6.6905\times10^{-1}, 6.9350\times10^{-1},$
$6.9350\times10^{-1}, 7.9592\times10^{-1}, 7.9592\times10^{-1}, 7.9940\times10^{-1},$
$7.9940\times10^{-1}, 3.9271\times10^{-1}$

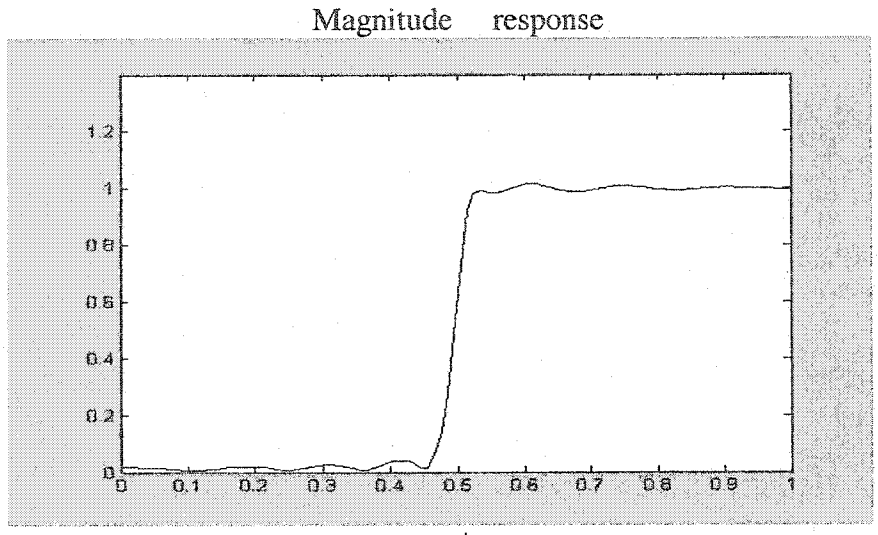


Figure 1.52: Magnitude of the desired filter (Example 1.10)

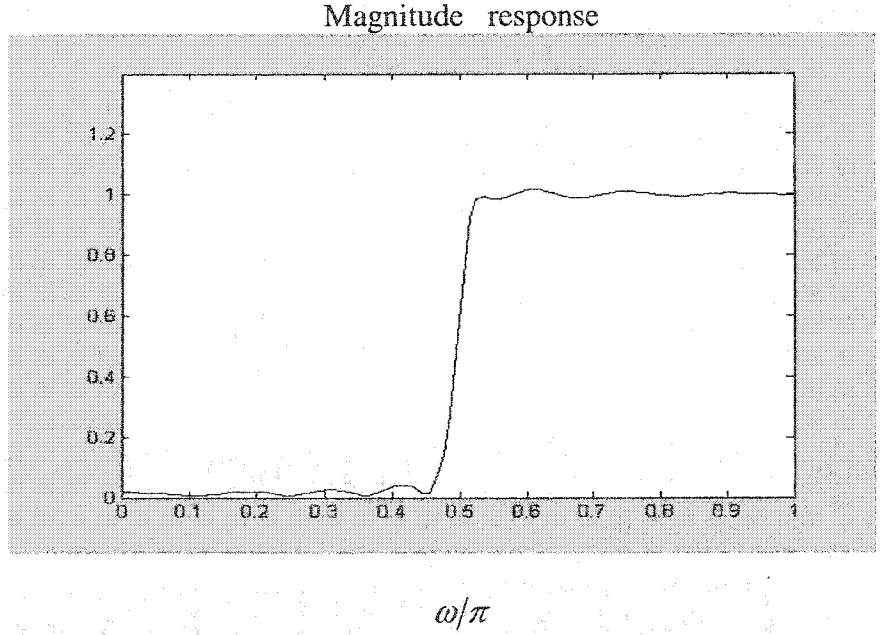


Figure 1.53: Magnitude of the proposed filter(max. pole radius=0.9276, Example 1.10)

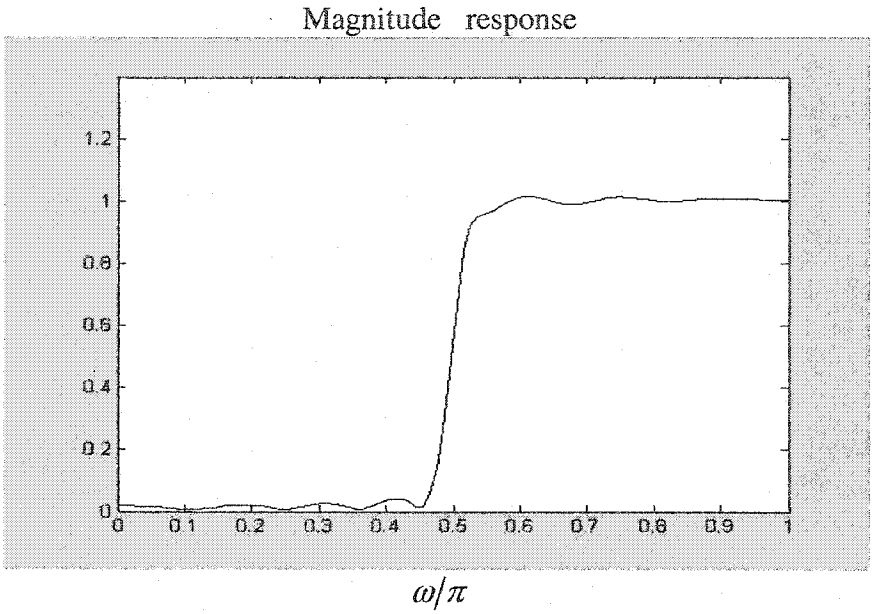


Figure 1.54: Magnitude of the proposed filter(max. pole radius=0.9200, Example 1.10)

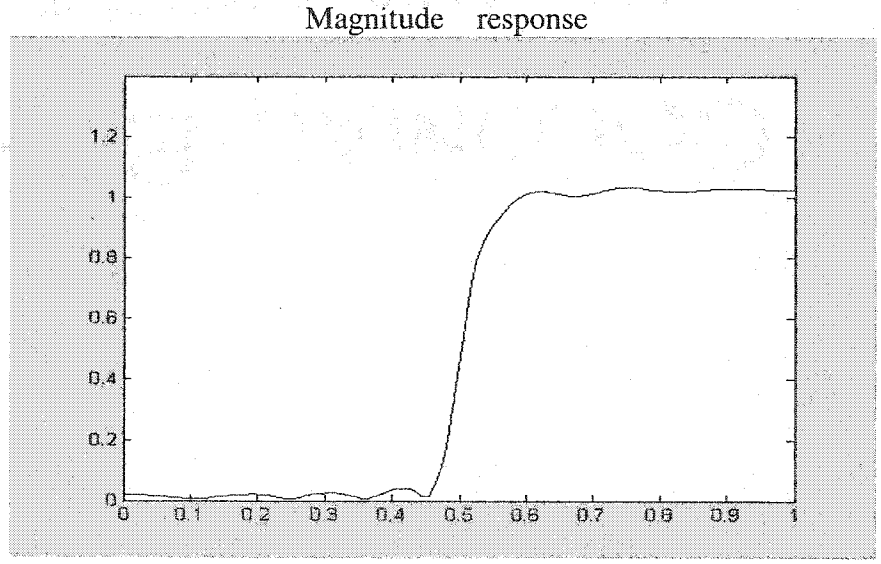


Figure 1.55: Magnitude of the proposed filter(max. pole radius=0.9000, Example 1.10)

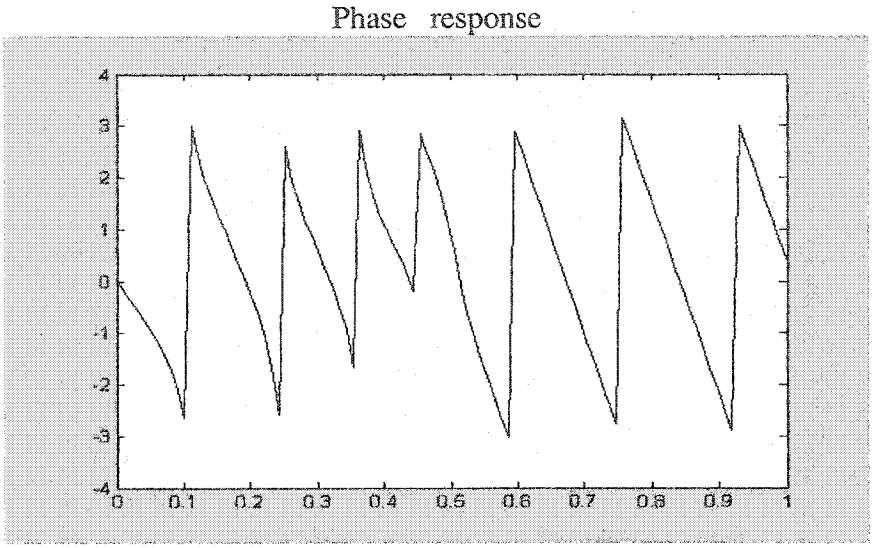


Figure 1.56: Phase of the desired filter (Example 1.10)

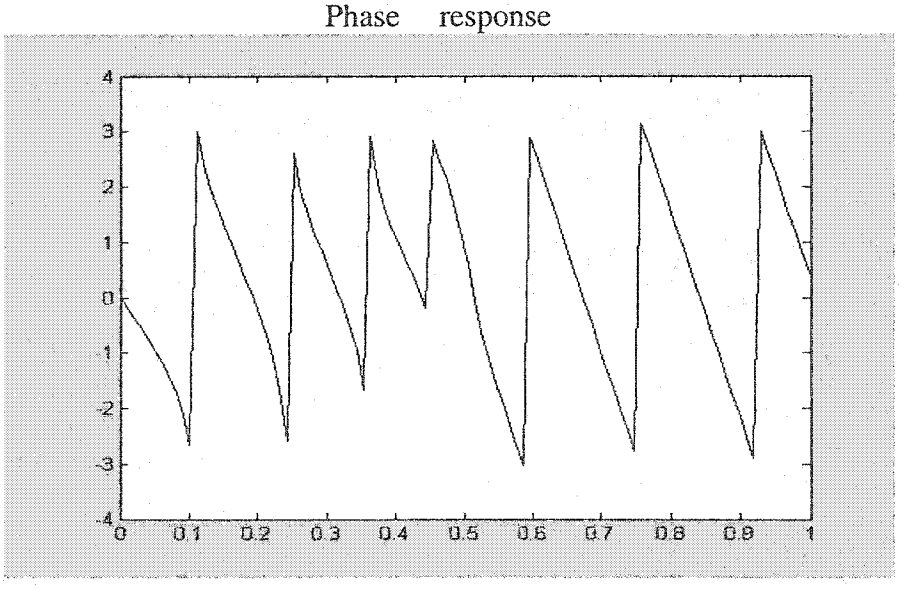


Figure 1.57: Phase of the proposed filter (max. pole radius=0.9276, Example 1.10)

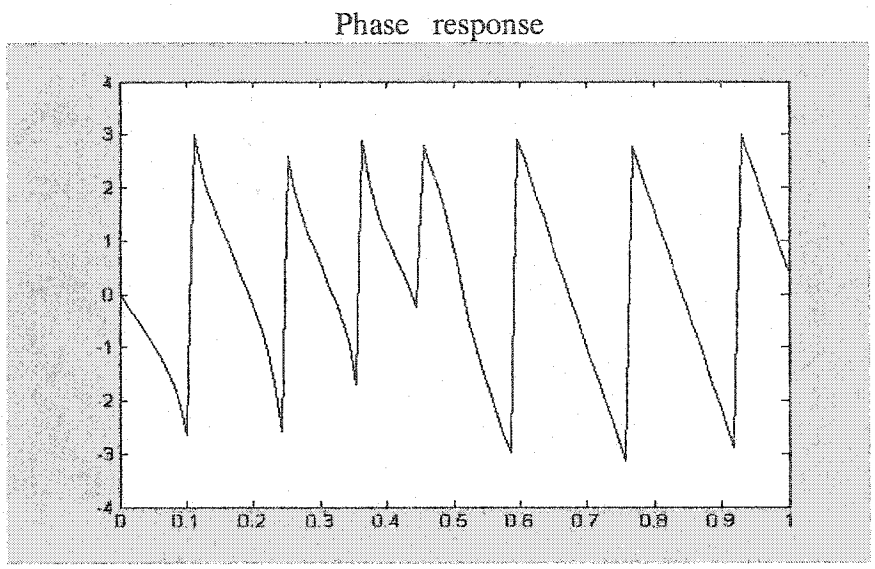


Figure 1.58: Phase of the proposed filter (max. pole radius=0.9200, Example 1.10)

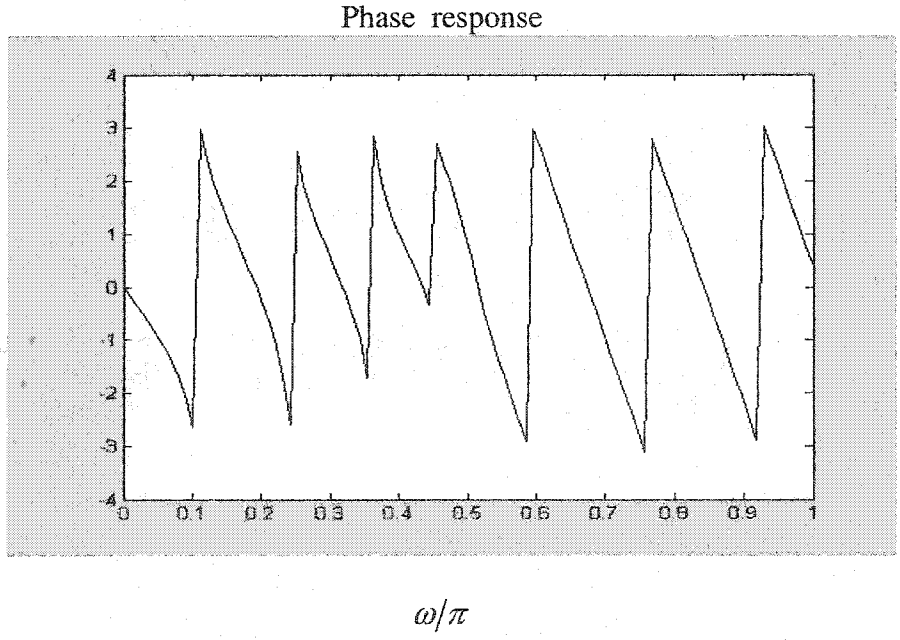


Figure 1.59: Phase of the proposed filter (max. pole radius=0.9000, Example 1.10)

The comparison results are shown in table 1.9.

Table 1.9 Frequency response error of example 1.10

	Gauss-Newton method	Proposed algorithm 2	Damped L-M method	Proposed algorithm 3
$E(x)$ (Max.radius: 0.9276)	9.8707 $\times 10^{-27}$	1.0520 $\times 10^{-26}$	1.5087 $\times 10^{-26}$	1.4188 $\times 10^{-25}$
$E(x)$ (Max.radius: 0.9200)	4.7935 $\times 10^{-2}$	3.7402 $\times 10^{-2}$	4.7775 $\times 10^{-2}$	3.7404 $\times 10^{-2}$
$E(x)$ (Max.radius: 0.9000)	1.1285	3.9469 $\times 10^{-1}$	1.1159	3.9470 $\times 10^{-1}$

- Example 1.11

This example is a fourteenth Chebyshev I low-pass filter, with 0.5 decibel of peak-to-peak ripple in the passband. The cutoff frequency is 0.6π. From Matlab function [b,a]=Cheby1(14,0.5,0.6), the desired filter coefficients b, a can be obtained. From Matlab function $[H_d(\omega)]$=freqz(b,a,100), the desired filter frequency response can be obtained.

The desired filter coefficients are:

$$b = 1.5658\times10^{-4}, 2.1921\times10^{-3}, 1.4249\times10^{-2},$$
$$5.6995\times10^{-2}, 1.5674\times10^{-1}, 3.1347\times10^{-1}, 4.7021\times10^{-1},$$
$$5.3738\times10^{-1}, 4.7021\times10^{-1}, 3.1347\times10^{-1}, 1.5674\times10^{-1},$$
$$5.6995\times10^{-2}, 1.4249\times10^{-2}, 2.1921\times10^{-3}, 1.5658\times10^{-4}.$$

a=1.0000, -1.3854, 4.3956, -5.9778, 9.5109, -1.1122×10^{1},
1.2350×10^{1}, -1.1682×10^{1}, 9.9109, -7.3472, 4.7655,
-2.6248, 1.2239, -4.1786×10^{-1}, 1.1836×10^{-1}.

The maximum pole radius of the disired filter is 0.9866. By using the *Proposed algorithm 3* , the proposed filters are designed. Three comparisons are made. The maximum pole radius of the proposed filter are set as 0.9866, 0.9800, 0.9500 respectively. Let the proposed filter order $M = N = 14$. Let the proposed filter frequency sampled points $L = 100$ between $0 \sim \pi$.

When the proposed filter maximum pole radius is 9.8660×10^{-1}, the proposed filter coefficients are:

$b = 1.5658\times10^{-4}$, 2.1921×10^{-3}, 1.4249×10^{-2},
5.6995×10^{-2}, 1.5674×10^{-1}, 3.1347×10^{-1}, 4.7021×10^{-1},
5.3738×10^{-1}, 4.7021×10^{-1}, 3.1347×10^{-1}, 1.5674×10^{-1},
5.6995×10^{-2}, 1.4249×10^{-2}, 2.1921×10^{-3}, 1.5658×10^{-4}.

a=1.0000, -1.3854, 4.3956, -5.9778, 9.5109, -1.1122×10^{1},
1.2350×10^{1}, -1.1682×10^{1}, 9.9109, -7.3472, 4.7655,
-2.6248, 1.2239, -4.1786×10^{-1}, 1.1836×10^{-1}.

The proposed filter poles radius:

7.1003×10^{-1}, 7.1003×10^{-1}, 7.6074×10^{-1}, 7.6074×10^{-1},
8.2587×10^{-1}, 8.2587×10^{-1}, 9.8659×10^{-1}, 9.8659×10^{-1},
9.5837×10^{-1}, 9.5837×10^{-1}, 9.2491×10^{-1}, 9.2491×10^{-1},
8.8187×10^{-1}, 8.8187×10^{-1}.

When the proposed filter maximum pole radius is 9.8000×10^{-1}, the proposed filter coefficients are:

$$b = 1.5658\times10^{-4}, 2.1921\times10^{-3}, 1.4249\times10^{-2},$$
$$5.6995\times10^{-2}, 1.5674\times10^{-1}, 3.1347\times10^{-1}, 4.7021\times10^{-1},$$
$$5.3738\times10^{-1}, 4.7021\times10^{-1}, 3.1347\times10^{-1}, 1.5674\times10^{-1},$$
$$5.6995\times10^{-2}, 1.4249\times10^{-2}, 2.1922\times10^{-3}, 1.5667\times10^{-4}.$$
$$a = 1.0000, -1.3895, 4.3908, -5.9710, 9.4789, -1.1071\times10^{1},$$
$$1.2272\times10^{1}, -1.1591\times10^{1}, 9.8187, -7.2688, 4.7088,$$
$$-2.5907, 1.2071, -4.1180\times10^{-1}, 1.1678\times10^{-1}.$$

The proposed filter poles radius:

$$7.1003\times10^{-1}, 7.1003\times10^{-1}, 7.6074\times10^{-1}, 7.6074\times10^{-1},$$
$$8.2587\times10^{-1}, 8.2587\times10^{-1}, 9.8000\times10^{-1}, 9.8000\times10^{-1},$$
$$9.5837\times10^{-1}, 9.5837\times10^{-1}, 9.2491\times10^{-1}, 9.2491\times10^{-1},$$
$$8.8187\times10^{-1}, 8.8187\times10^{-1}.$$

When the proposed filter maximum pole radius is 9.5000×10^{-1}, the proposed filter coefficients are:

$$b = 1.5653\times10^{-4}, 2.1922\times10^{-3}, 1.4248\times10^{-2},$$
$$5.6995\times10^{-2}, 1.5673\times10^{-1}, 3.1347\times10^{-1}, 4.7020\times10^{-1},$$
$$5.3737\times10^{-1}, 4.7020\times10^{-1}, 3.1347\times10^{-1}, 1.5673\times10^{-1},$$
$$5.6994\times10^{-2}, 1.4250\times10^{-2}, 2.1926\times10^{-3}, 1.5783\times10^{-4}.$$

$$a = 1.0000, -1.4126, 4.3626, -5.9309, 9.2958, -1.0779\times10^{1},$$

1.1831×10^{1}, -1.1077×10^{1}, 9.3005, -6.8281, 4.3902,

-2.3987, 1.1128, -3.7755×10^{-1}, 1.0783×10^{-1}.

The proposed filter poles radius:

7.1003×10^{-1}, 7.1003×10^{-1}, 7.6074×10^{-1}, 7.6074×10^{-1},

8.2587×10^{-1}, 8.2587×10^{-1}, 9.4999×10^{-1}, 9.4999×10^{-1},

9.5000×10^{-1}, 9.5000×10^{-1}, 9.2491×10^{-1}, 9.2491×10^{-1},

8.8187×10^{-1}, 8.8187×10^{-1}.

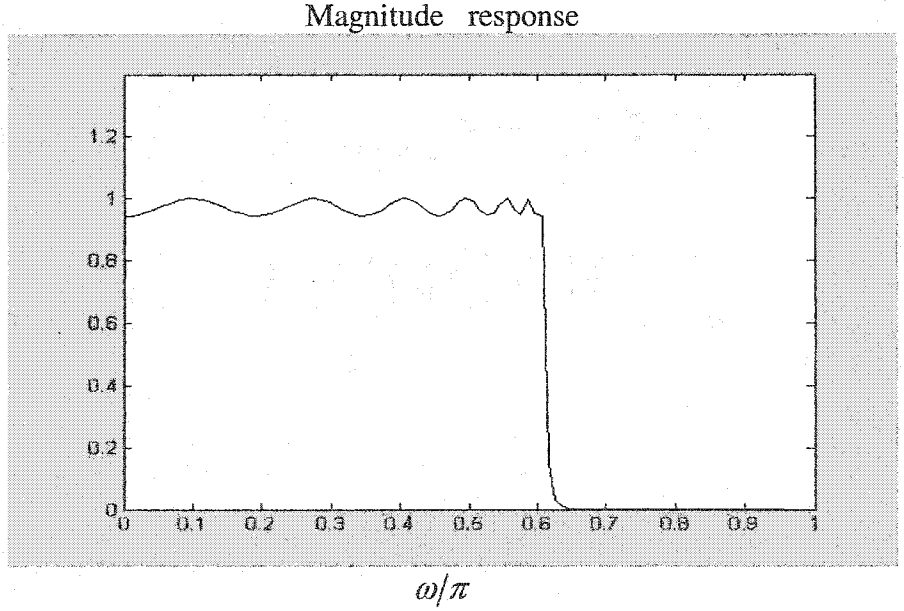


Figure 1.60: Magnitude of the desired filter (Example 1.11)

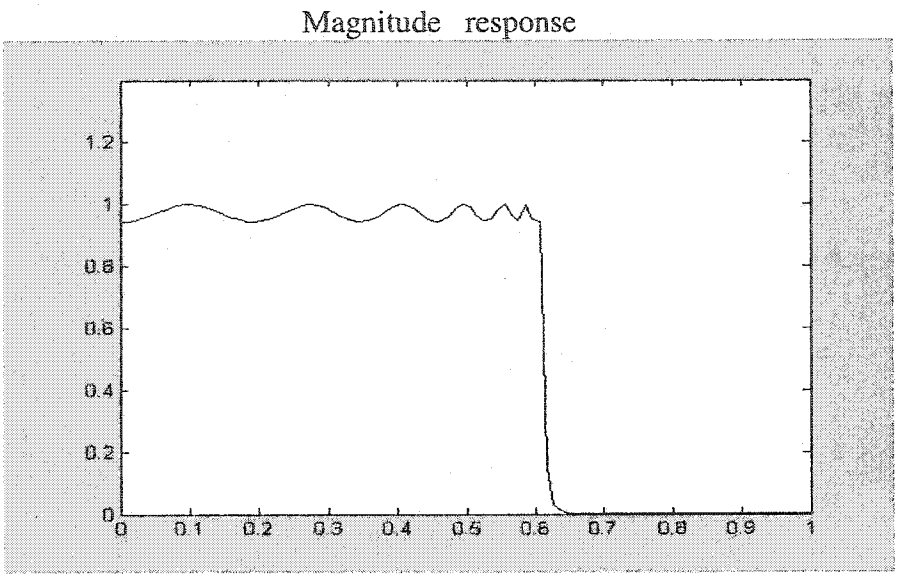


Figure 1.61: Magnitude of the proposed filter (max. pole radius=0.9866, Example 1.11)

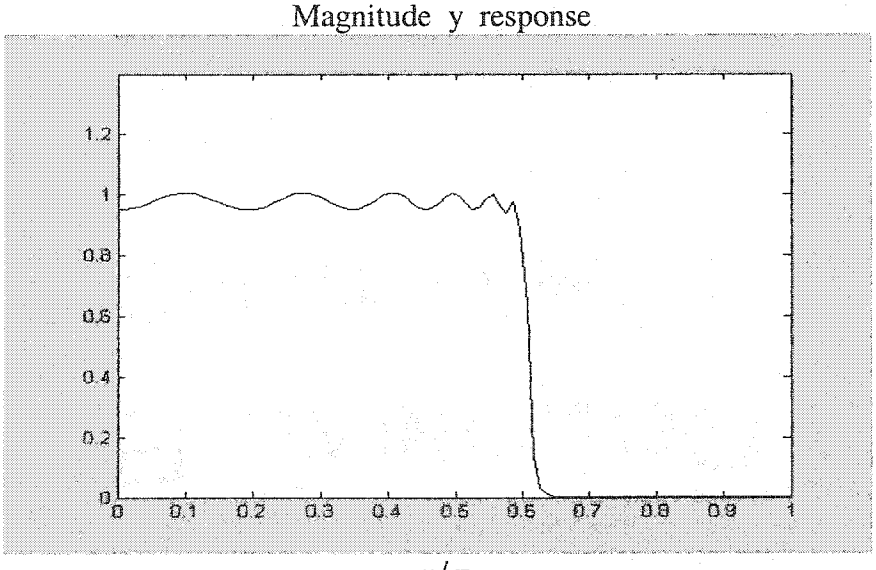


Figure 1.62: Magnitude of the proposed filter (max. pole radius=0.9800, Example 1.11)

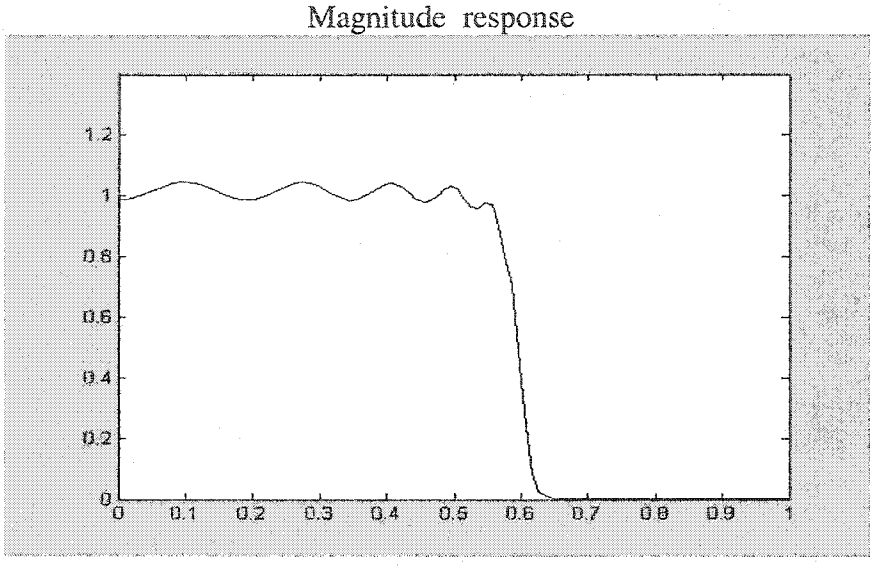


Figure 1.63: Magnitude of the proposed filter (max. pole radius=0.9500, Example 1.11)

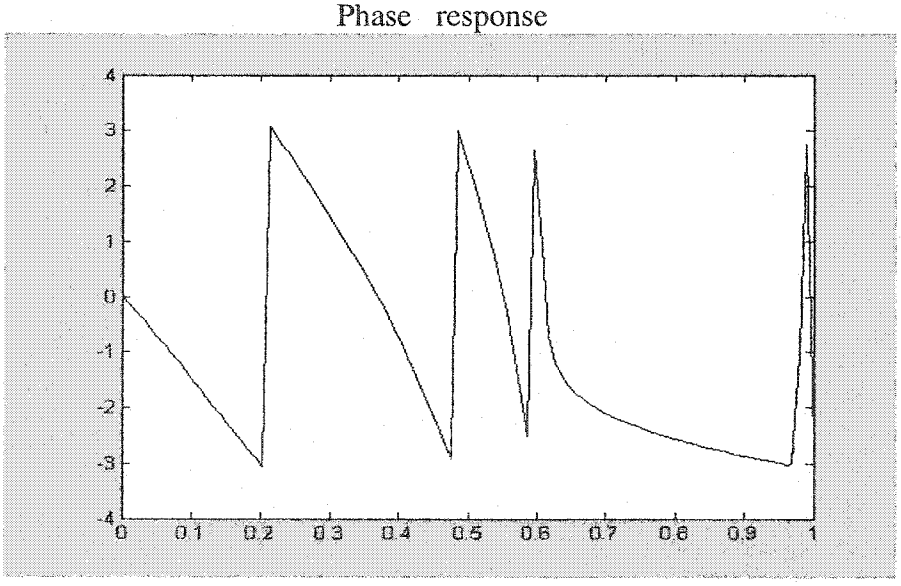


Figure 1.64: Phase of the desired filter (Example 1.11)

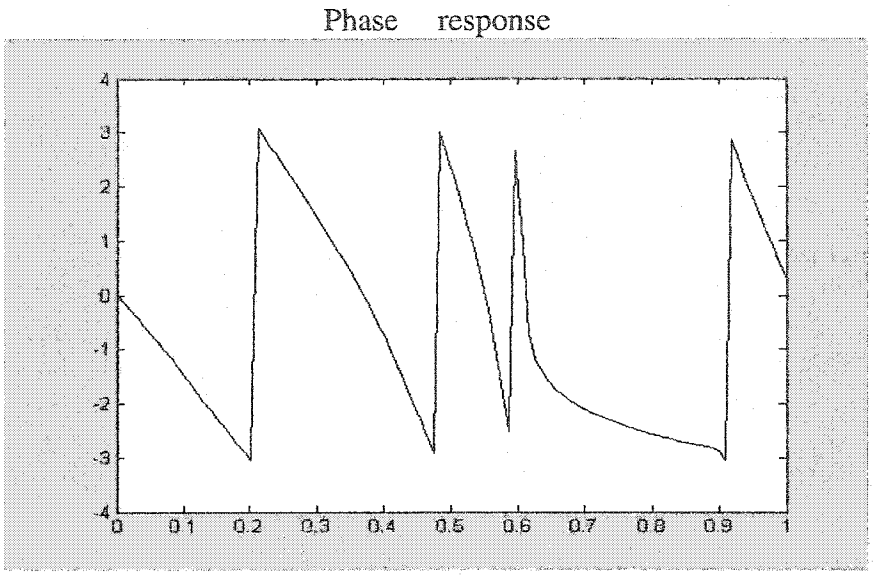


Figure 1.65: Phase of the proposed filter (max. pole radius=0.9866, Example 1.11)

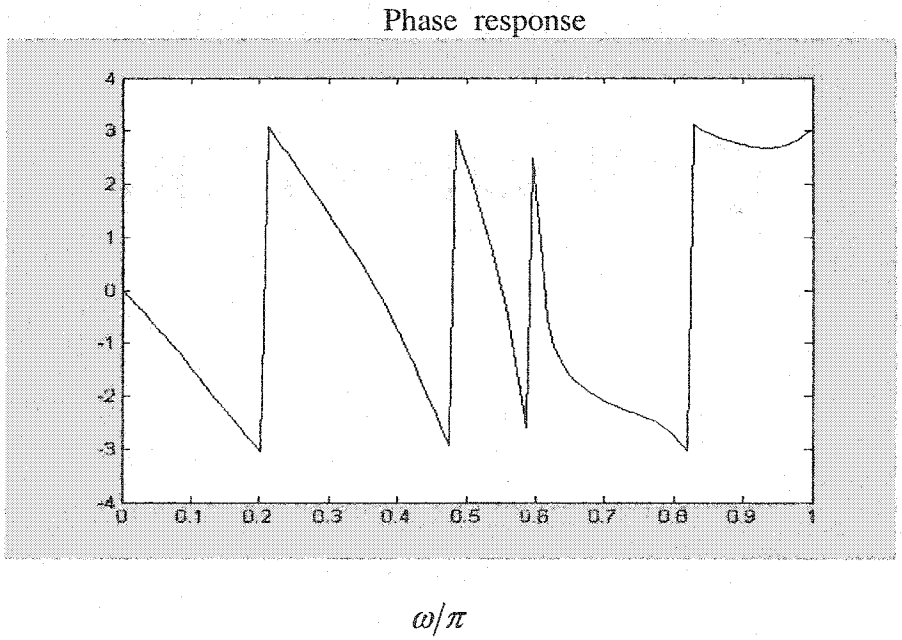


Figure 1.66: Phase of the proposed filter (max. pole radius=0.9800, Example 1.11)

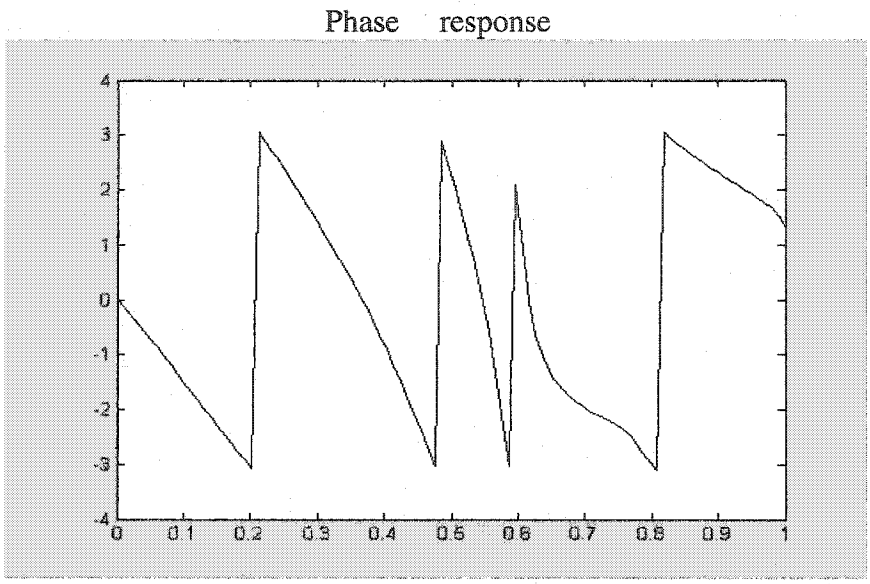


Figure 1.67: Phase of the proposed filter (max. pole radius=0.9500, Example 1.11)

The comparison results are shown in table 1.10.

Table 1.10 Frequency response error of example 1.11

	Gauss-Newton method	Proposed algorithm 2	Damped L-M method	Proposed algorithm 3
$E(x)$ (Max.radius:0.9866)	1.9771×10^{-22}	2.4390×10^{-22}	6.5858×10^{-17}	5.6104×10^{-15}
$E(x)$ (Max.radius:0.9800)	9.4489	1.4922×10^{-1}	8.8425×10^{-2}	1.4922×10^{-1}
$E(x)$ (Max.radius:0.9500)	1.6651×10^{1}	1.9476	3.7927	1.9470

In this chapter, three algorithms are proposed to design IIR filters in the frequency domain. *Proposed algorithm 1* approximates the arbitrary magnitude response of the desired filter. *Proposed algorithm 2* and *Proposed algorithm 3* approximate simultaneously the arbitrary magnitude response and phase response of the desired filter with specified stability region. Several design examples are tested to compare the proposed algorithms with other algorithms. The comparison results show that the proposed algorithms have smaller error than other algorithms. *Proposed algorithm 1* and *Proposed algorithm 2* have good convergence.

CHAPTER II

LEAST SQUARES DESIGNS OF IIR FILTERS IN THE TIME DOMAIN

2.1 Proposed algorithm 4 (QR-SVD-PRONY algorithm)

In this chapter, least squares procedures are utilized to design IIR filters with an arbitrary time impulse response. An algorithm (QR-SVD-PRONY algorithm) is proposed based on some numerical analysis techniques and Prony method [T.W.Parks, (1987)]. Several examples are tested to compare this proposed algorithm with *Matlab* Prony( ) function. The comparison results show that this proposed algorithm has good results for designing IIR filters in the time-domain.

The transfer function of an IIR filter is given by:

$$H(z)=\frac{B(z)}{A(z)}=\frac{b_0+b_1z^{-1}+b_2z^{-2}+...+b_Mz^{-M}}{1+a_1z^{-1}+a_2z^{-2}+...+a_Nz^{-N}} \tag{2.1}$$

and $H(z)$ can be written as another form:

$$H(z)=\sum_{n=0}^{\infty}h(n)z^{-n}.$$

where $h(n)$ is the impulse response related to $H(z)$. Equation (2.1) can be written as

$$B(z)=H(z)A(z) \tag{2.2}$$

Equation (2.2) can be expressed as the convolution of $h(n)$ and a_j,

$$b_i=h(n)*a_j \qquad (i=0,1,2,...,M,\ \ j=0,1,2,3,...,N) \tag{2.3}$$

This convolution can be rewritten as a matrix form. Taking the first $k+1$ terms of the impulse response $h(n)$ [T.W.Parks,(1987)]. That is

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ : \\ : \\ b_M \\ 0 \\ : \\ 0 \end{pmatrix} = \begin{pmatrix} h_0 & 0 & 0 & ..... & 0 \\ h_1 & h_0 & 0 & : & : \\ h_2 & h_1 & h_0 & : & : \\ h_3 & h_2 & h_1 & : & : \\ : & h_3 & h_2 & : & : \\ h_M & : & h_3 & : & : \\ h_{M+1} & : & : & : & : \\ : & : & : & : & : \\ h_k & h_{k-1} & h_{k-2} & : & h_{k-N} \end{pmatrix} \begin{pmatrix} 1 \\ a_1 \\ a_2 \\ : \\ : \\ a_N \end{pmatrix} \tag{2.4}$$

This equation is partitioned such that

$$\begin{pmatrix} b \\ --- \\ 0 \end{pmatrix} = \begin{pmatrix} H_1 \\ ------ \\ h^* \mid H_2 \end{pmatrix} \begin{pmatrix} 1 \\ --- \\ a^* \end{pmatrix} \tag{2.5}$$

where b is the $(M+1)\times 1$ numerator coefficients vector of equation (2.1), H_1 is the $(M+1)\times(N+1)$ partition matrix of equation (2.4), $h^* = [h_{M+1}, h_{M+2}, ..., h_k]^T$, H_2 is the $(k-M)\times N$ remaining partition matrix, and a^* is the vector of the N denominator coefficients $(a_0 = 1)$. The lower $k-M$ equations of (2.5) can be expressed as

$$0 = h^* + H_2 a^* \quad \text{or} \quad H_2 a^* = -h^* \tag{2.6}$$

Then a^* can be solved from equation (2.6), i.e., the denominator coefficients of equation (2.1) can be obtained. The upper $M+1$ equations of (2.5) can be expressed as:

$$b = H_1 a \tag{2.7}$$

the numerator coefficients of equation (2.1) can be calculated from equation (2.7) [T.W.Parks, (1987)].

If $k = M + N$, H_2 is square. If H_2 is not singular , the coefficients a can be solved from equation (2.6), and b can be solved from equation (2.7). If H_2 is singular, equation (2.6) may have many solutions [T.W.Parks, (1987)].

If $k > M + N$, as is usually the case, equation (2.6) is an overdetermined system, that is, it has more equations than unknowns. Thus we can not expect to find an a^* that satisfies equation (2.6) exactly. Instead we may seek an a^* for which the sum of the squares of the residuals $r = H_2 a^* + h_1$ is minimized.

The least squares problem for the system equation (2.6) is to find a^* for which $\|r\|_2$ is minimized, where $r = H_2 a^* + h_1$ is the vector of residuals. That is, $\|H_2 a^* + h_1\|_2$ is minimized. To minimize $\|r\|_2 = \|H_2 a^* + h_1\|_2$, the traditional method is to solve the normal equation

$$H_2{}^T H_2 a^* = -H_2{}^T h_1 \tag{2.8}$$

If H_2 has full rank, then

$$a^* = -(H_2{}^T H_2)^{-1} H_2{}^T h_1 \tag{2.9}$$

Numerical analysis shows that the computation of inverse of a matrix should be avoided. To compute the inverse of a matrix is not only very inefficient but also very inaccurate.

In the 70's and 80's, Cholesky factorization was often used to solve normal equation (2.8). $H_2{}^T H_2$ can be factorized into Cholesky factor L and L^T, such that $H_2{}^T H_2 = LL^T$, where L is an $N \times N$ lower triangular matrix, and L^T is the transpose of L. According to equation (2.8), $H_2{}^T H_2 a^* = LL^T a^* = -H_2{}^T h^*$. Let $y = L^T a^*$, then solve y from $Ly = -H_2{}^T h^*$. Then solve a^* from $L^T a^* = y$. Hence, a^* can be obtained by using the Cholesky factorization of $H_2{}^T H_2$.

The advantage of Cholesky factorization is that the computation of Cholesky factorization is simple. But, there are some disadvantages. Using Cholesky factorization, $H_2{}^T H_2$ has to be formed, then find the Cholesky factorization of $H_2{}^T H_2$. But $H_2{}^T H_2$ is often numerically ill conditioned, the solution of a^* maybe not accurate. Therefore, in practice, the direct Cholesky factorization of H_2 is usually avoided.

Instead, the factorization can be obtained by directly operating on H_2 using the QR factorization, which can provide more accurate solutions and efficient algorithms. If H_2 has full column rank, i.e., $rank(H_2) = N$, H_2 can be factorized by QR factorization. According to the definition of QR factorization, QR factorization on H_2 gives:

$$Q^T H_2 = \begin{bmatrix} R \\ 0 \end{bmatrix} \tag{2.10}$$

where R is an $N \times N$ triangular matrix, Q is a $(k-M)\times(k-M)$ orthogonal matrix. Since Q is orthogonal, then $Q^T Q = QQ^T = I$, where I is $(k-M)\times(k-M)$ identity matrix. Equation (2.10) is multiplied by Q on both sides to give

$$QQ^T H_2 = Q\begin{bmatrix} R \\ \\ 0 \end{bmatrix} \tag{2.11}$$

Since $QQ^T = I$, equation (2.11) is rewritten as

$$H_2 = Q\begin{bmatrix} R \\ \\ 0 \end{bmatrix} = [Q_1 \quad Q_2]\begin{bmatrix} R \\ \\ 0 \end{bmatrix} = Q_1 R \tag{2.12}$$

where Q_1 is a $(k-M)\times N$ orthogonal matrix, i.e., $Q_1^T Q_1 = I$.

For orthogonal matrix Q and vector x, numerical algebra shows that the following equation exists,

$$\|Qx\|_2 = ((Qx)^T Qx)^{\frac{1}{2}} = (x^T Q^T Qx)^{\frac{1}{2}} = (x^T I x)^{\frac{1}{2}} = (x^T x)^{\frac{1}{2}} = \|x\|_2$$

where I is the identity matrix. The equation above shows that the L_2 norm of the product of an orthogonal matrix and a vector equals the L_2 norm of the vector.
That is

$$\|Qx\|_2 = \|x\|_2 \tag{2.13}$$

Since $\|r\|_2 = \|H_2 a^* + h^*\|_2$, then according to equation (2.13), equation (2.14) exists.

$$\|r\|_2^{\,2} = \|H_2 a^* + h_1\|_2^{\,2} = \|Q^T(H_2 a^* + h_1)\|_2^{\,2} = \|Q^T H_2 a^* + Q^T h_1\|_2^{\,2} \tag{2.14}$$

Partitioning Q into Q_1, Q_2 yields

$$Q = [Q_1 \quad Q_2]$$

where Q_1 is a $(k-M)\times N$ orthogonal matrix, Q_2 is a $(k-M)\times(k-M-N)$ orthogonal matrix. The transpose of Q gives

$$Q^T = [Q_1 \quad Q_2]^T = \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \tag{2.15}$$

Substituting equation (2.10) and (2.15) into equation (2.14) yields

$$\|r\|_2^2 = \|Q^T H_2 a^* + Q^T h_1\|_2^2 = \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} a^* + Q^T h_1 \right\|_2^2 = \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} a^* + \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} h_1 \right\|_2^2$$

$$= \left\| \begin{bmatrix} Ra^* \\ 0 \end{bmatrix} + \begin{bmatrix} Q_1^T h_1 \\ Q_2^T h_1 \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} Ra^* + Q_1^T h_1 \\ 0 + Q_2^T h_1 \end{bmatrix} \right\|_2^2$$

$$= \|Ra^* + Q_1^T h_1\|_2^2 + \|Q_2^T h_1\|_2^2$$

In order to minimize $\|r\|_2$, $\|Ra^* + Q_1^T h_1\|_2$ and $\|Q_2^T h_1\|_2$ should be minimized. If $\|Ra^* + Q_1^T h_1\|_2 = 0$, then $\min\|r\|_2 = \|Q_2^T h_1\|_2$. Therefore, let $Ra^* + Q_1^T h_1 = 0$, equation (2.16) exists

$$Ra^* = -Q_1^T h_1 \tag{2.16}$$

The coefficients a^* can be solved from equation (2.16) by using back substitution. After obtaining a, the coefficients b can be obtained from equation (2.7).

If H_2 is a rank-deficient matrix, i.e., H_2 does not have full column rank, QR factorization can not be used to solve equation (2.6). Instead, singular value decomposition (SVD) should be used to solve rank-deficient least squares problems. SVD decomposition shows that for any matrix H_2, there exists orthogonal matrix U and orthogonal matrix V such that

$$U^T H_2 V = \Sigma = \begin{bmatrix} \Sigma_1 \\ \\ 0 \end{bmatrix}, \qquad \Sigma_1 = diag(\sigma_1, \sigma_2, \sigma_3, \sigma_4, ..., \sigma_N)$$

with $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq ..... \geq \sigma_t > \sigma_{t+1} = ... = \sigma_N = 0$, $t = rank(H_2)$, H_2 is $(k-M)\times N$ matrix, U is $(k-M)\times(k-M)$ orthogonal matrix, V is $N \times N$ orthogonal matrix. Since $UU^T = I$ and $VV^T = I$, then

$$H_2 = U\Sigma V^T \tag{2.17}$$

Equation (2.17) is the SVD decomposition of H_2, and $\sigma_1, \sigma_2, \sigma_3, ..., \sigma_t, ..., \sigma_N$ are the singular values of H_2. SVD decomposition of H_2 can be used to minimize $\|r\|_2$. According to equation (2.13), for a orthogonal matrix U^T,

$$\|r\|_2 = \|H_2 a^* + h_1\|_2 = \|U^T (H_2 a^* + h_1)\|_2 .$$

Substituting equation (2.17) into the above equation, the following equation exists

$$\begin{aligned} \|r\|_2 &= \|H_2 a^* + h_1\|_2 = \|U^T (H_2 a^* + h_1)\|_2 \\ &= \|U^T U \Sigma V^T a^* + U^T h_1\|_2 = \|\Sigma V^T a^* + U^T h_1\|_2 \end{aligned} \tag{2.18}$$

Since $rank(H_2)=t$, Σ is partitioned such that $\Sigma=\begin{bmatrix}\Sigma_1 & 0\\ 0 & 0\end{bmatrix}$, where $\Sigma_1=diag(\sigma_1,\sigma_2,\sigma_3,...,\sigma_t)$, $U=[\underbrace{U_1}_{t} \quad U_2]$, $V=[\underbrace{V_1}_{t} \quad V_2]$, U_1 is a $(k-M)\times t$ matrix, and V_1 is a $N\times t$ matrix. Substituting the above Σ,U,V into equation (2.18), the following equations exist.

$$\|r\|_2=\|H_2a^*+h_1\|_2^2=\left\|\begin{bmatrix}\Sigma_1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1^T\\ V_2^T\end{bmatrix}a^*+\begin{bmatrix}U_1^T\\ U_2^T\end{bmatrix}h_1\right\|_2^2$$

$$=\left\|\begin{bmatrix}\Sigma_1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1^Ta^*\\ V_2^Ta^*\end{bmatrix}+\begin{bmatrix}U_1^Th_1\\ U_2^Th_1\end{bmatrix}\right\|_2^2=\left\|\begin{matrix}\Sigma_1V_1^Ta^*+U_1^Th_1\\ U_2^Th_1\end{matrix}\right\|_2^2$$

$$=\|\Sigma_1V_1^Ta^*+U_1^Th_1\|_2^2+\|U_2^Th_1\|_2^2$$

If $\Sigma_1V_1^Ta^*+U_1^Th_1=0$, then $\min\|r\|_2=\min\|H_2a^*+h_1\|_2=\|U_2^Th_1\|_2$. Hence,

$$\Sigma_1V_1^Ta^*=-U_1^Th_1 \tag{2.19}$$

Since V_1 is orthogonal, then $V_1V_1^T=I$. Equation (2.19) is multiplied by $V_1\Sigma_1^{-1}$ on both sides, we have

$$V_1\Sigma_1^{-1}\Sigma_1V_1^Ta^*=-V_1\Sigma_1^{-1}U_1^Th_1$$

i.e., $$V_1IV_1^Ta^*=-V_1\Sigma_1^{-1}U_1^Th_1$$

$$V_1V_1^Ta^*=-V_1\Sigma_1^{-1}U_1^Th_1$$

$$a^*=-V_1\Sigma_1^{-1}U_1^Th_1 \tag{2.20}$$

where $\Sigma_1^{-1}=diag(\frac{1}{\sigma_1},\frac{1}{\sigma_2},\frac{1}{\sigma_3},.....,\frac{1}{\sigma_t})$.

Therefore, when H_2 is rank-deficient, *SVD* of H_2 can be used to obtain a^* by solving equation (2.20). After obtaining the denominator coefficients a, the numerator coefficients b can be obtained from equation (2.7). Thus, we have proposed an algorithm, we call it *Proposed algorithm 4 (QR-SVD-PRONY* algorithm). That is, when H_2 is full rank, the *QR* factorization of H_2 is used to obtain the coefficients a. When H_2 is rank-deficient, the *SVD* of H_2 is used to obtain the coefficients a.

The error, which is used to test the performance of *Proposed algorithm 4*, is defined as the sum of the squares of the difference between the actual impulse response and the desired impulse response.

$$e(n) = \sum_{n=0}^{k} (h(n) - h_d(n))^2 \tag{2.21}$$

There is a *Matlab* function *prony*( ) [Natic Mass, (1994)], using *QR* factorization with pivoting to design IIR filter. In the following section, several examples are used to compare *Proposed algorithm 4* and *Matlab* function *prony*( ).

2.2 Examples

- Example 2.1

The truncated desired time impulse response of an IIR system is given by:

$$h_d(n) = (0.9\cos\frac{n\pi}{3} + 0.6\cos\frac{n\pi}{4})R_N(n) \tag{2.22}$$

The window size is chosen as 10. Let the proposed filter order $M = N = 4$.

The truncated desired time impulse response:

$h_d(n)$=1.5000, 8.7426×10^{-1}, -4.5000×10^{-1}, -1.3243, -1.0500, 2.5736×10^{-2}, 9.0000×10^{-1}, 8.7426×10^{-1}, 1.5000×10^{-1}, -4.7574×10^{-1}.

By using the *Proposed algorithm 4*, the proposed filter is designed.

The proposed filter time impulse response:

$h(n)$ =1.5000, 8.7426×10^{-1}, -4.5000×10^{-1}, -1.3243, -1.0500, 2.5736×10^{-2}, 9.0000×10^{-1}, 8.7426×10^{-1}, 1.5000×10^{-1}, -4.7574×10^{-1}.

The proposed filter coefficients are:

b =1.5000, -2.7471, 2.5607, -8.7426×10^{-1}, -4.4409×10^{-16}.
a =1.0000, -2.4142, 3.4142, -2.4142, 1.0000.

The proposed filter time impulse response error is:

$e(n)$ =1.6175×10^{-29}.

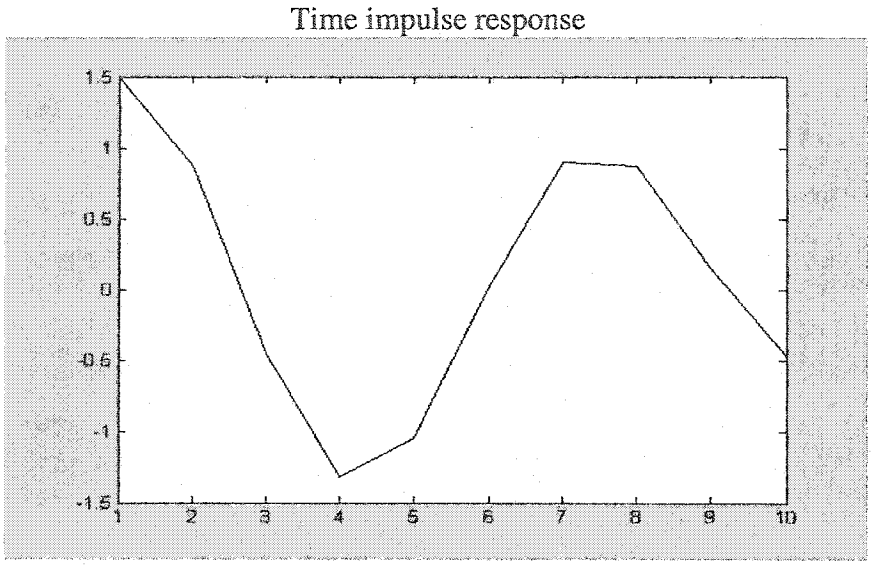


Figure 2.1: Time impulse response of the desired filter (Example 2.1)

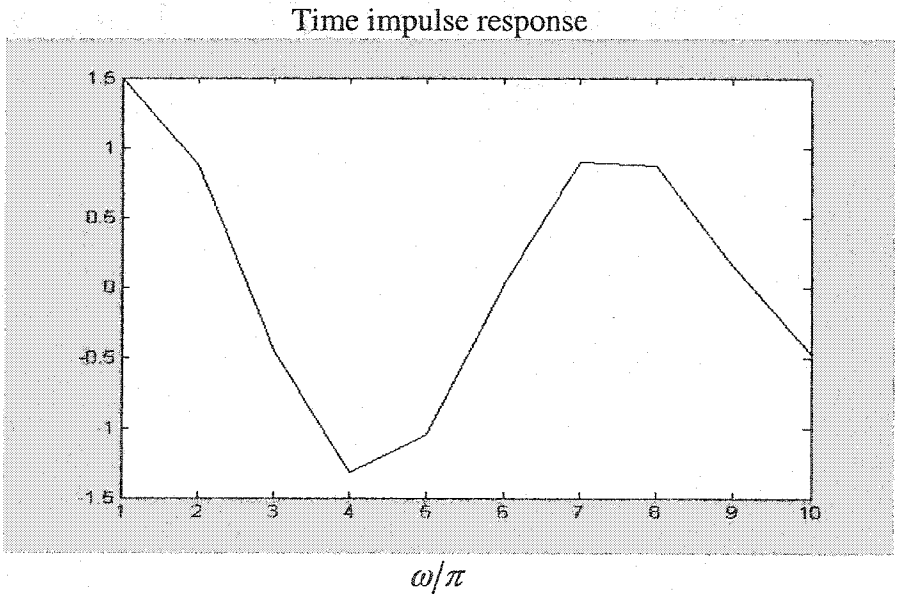


Figure 2.2: Time impulse response of the proposed filter (Example 2.1)

- Example 2.2

The truncated desired time impulse response of an IIR system in [Michael J.Corinthios, (1996)] is given by:

$$h_d(n) = \sum_{i=1}^{P} A_i e^{-\alpha_i n} \cos(\beta_i n + \theta_i) R_N(n) \tag{2.23}$$

The parameters of the time sequence are given by:

$$p = 7, \quad A_i = 1; \quad i = 1,2,...,6,7, \quad \alpha_1 = \theta_1 = 0,$$

$$\alpha_i = \alpha_{i-1} + \Delta\alpha, \quad \theta_i = \theta_{i-1} + \Delta\theta; \quad i = 2,3,...,7.$$

$$\Delta\alpha = 0.005, \quad \Delta\theta = \frac{\pi}{28},$$

$$\beta_i = (2\pi/N) \cdot m_i, \text{ where}$$

$$m_i = 415, 87, 169, 5, 251, 497, 333, \quad \text{for } i = 1, 2, 3, ..., 7,$$

respectively.

The window size is chosen as 1024, i.e., 1024 values of $h(n)$ and $h_d(n)$ are chosen to compute equation (2.21). Similarly, three comparisons are made.

a) Let the order of the numerator polynomial in equation (2.1) $M = 13$, the order of the denominator polynomial in equation (2.1) $N = 14$. In this case, the matrix H_2 is full rank.

b) Let $M = 14$, $N = 15$, in this case, the matrix H_2 is rank-deficient.

c) Let $M = 15$, $N = 16$, in this case, the matrix H_2 is rank-deficient.

By using the *Proposed algorithm 4*, the proposed filters are designed.

When the proposed filter order $M = 13$, $N = 14$, the proposed filter coefficients are:

b =6.4420, -2.6937, -3.4334, 1.5798, -1.0247, 5.0081×10^{-2}, 3.1124×10^{-1},

-4.8361×10^{-1}, 2.8552×10^{-1}, 1.6954×10^{-2}, -7.0732×10^{-1}, 1.2350,
-1.3079×10^{-1}, -1.3077.

a=1.0000, -2.6839×10^{-1}, -7.3316×10^{-1}, 1.4396×10^{-1}, -4.2853×10^{-2}, -2.9805×10^{-2},
-1.9190×10^{-2}, 3.5158×10^{-2}, -1.0163×10^{-1}, 1.0624×10^{-1}, -1.5618×10^{-1},
5.6129×10^{-2}, -6.0359×10^{-1}, -1.7361×10^{-1}, 8.1058×10^{-1}.

When the proposed filter order $M = 14$, $N = 15$, the proposed filter coefficients are:

b=6.4420, -2.0067, -3.7206, 1.2137, -8.5619×10^{-1}, -5.9193×10^{-2}, 3.1658×10^{-1},
-4.5042×10^{-1}, 2.3394×10^{-1}, 4.7402×10^{-2}, -7.0551×10^{-1}, 1.1595,
9.0790×10^{-4}, -1.3216, -1.3945×10^{-1}.

a=1.0000, -1.6175×10^{-1}, -7.6179×10^{-1}, 6.5769×10^{-2}, -2.7501×10^{-2}, -3.4374×10^{-2},
-2.2369×10^{-2}, 3.3111×10^{-2}, -9.7878×10^{-2}, 9.5403×10^{-2}, -1.4485×10^{-1},
3.9473×10^{-2}, -5.9761×10^{-1}, -2.3798×10^{-1}, 7.9207×10^{-1}, 8.6443×10^{-2}.

When the proposed filter order $M = 15$, $N = 16$, the proposed filter coefficients are:

b=6.4420, -1.7137, -1.0959, -9.1208×10^{-2}, -2.2485, 5.6795×10^{-1}, -1.1812×10^{-1},
-4.1491×10^{-1}, 3.4468×10^{-1}, -1.4585×10^{-1}, -5.8298×10^{-1}, 1.1346, -2.4457×10^{-1},
-8.0089×10^{-1}, -2.5470×10^{-1}, -5.5766×10^{-1}.

a=1.0000, -1.1627×10^{-1}, -3.4753×10^{-1}, -8.2033×10^{-2}, -3.3362×10^{-1}, 2.5068×10^{-2},
-4.1999×10^{-2}, 1.9528×10^{-2}, -1.0446×10^{-1}, 1.0577×10^{-1}, -1.8336×10^{-1},
7.6777×10^{-2}, -6.6166×10^{-1}, -2.4150×10^{-1}, 5.2677×10^{-1}, $.9028\times10^{-2}$,
3.4568×10^{-1}.

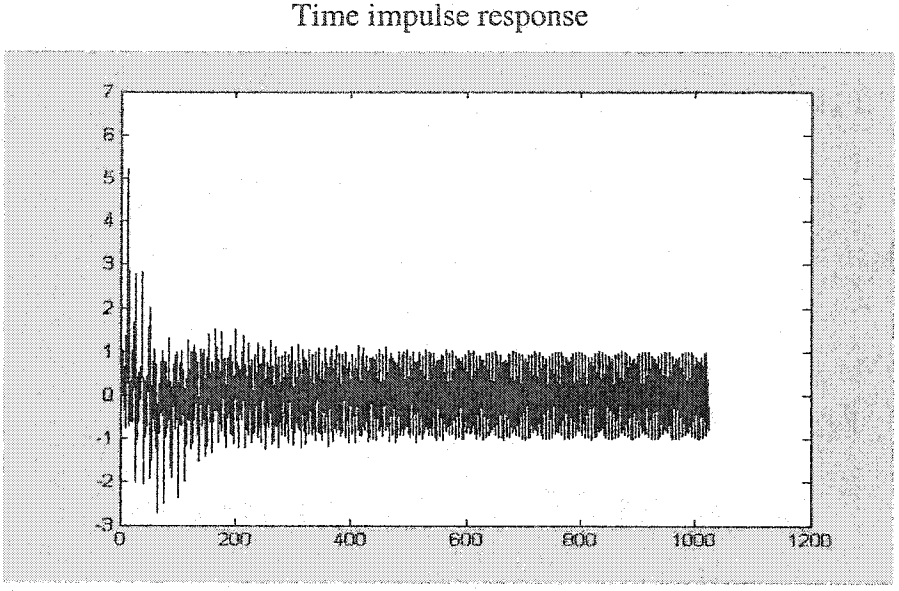


Figure 2.3: Time impulse response of the desired filter (Example 2.2)

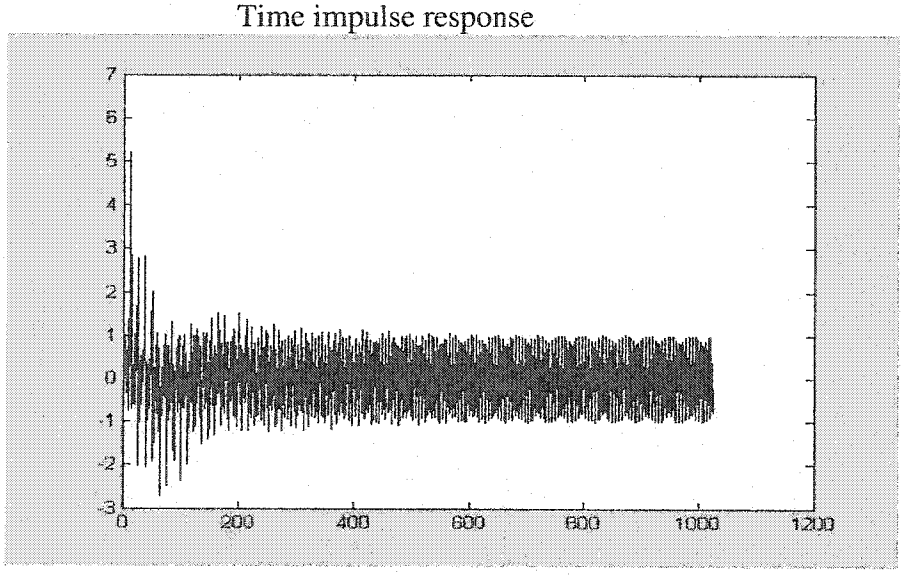


Figure 2.4: Time impulse response of the proposed filter ($M = 13, N = 14$, Example 2.2)

The comparison results $e(n)$ in equation (2.21) are shown in the following table.

Table 2.1 The comparison results of example 2.2

	Proposed algorithm 4	Matlab prony( ) function
$e(n)$ $(M=13, N=14)$	2.9521×10^{-24}	3.3527×10^{-24}
$e(n)$ $(M=14, N=15)$	2.9900×10^{-24}	7.2237×10^{-24}
$e(n)$ $(M=15, N=16)$	2.9126×10^{-24}	3.9022×10^{-24}

Table 2.1 shows that the errors $e(n)$ of the proposed algorithm is a little smaller than the errors of *Matlab prony*() function.

- Example 2.3

The truncated desired time impulse response of an IIR system in [Michael J.Corinthios, (1996)] is given by:

$$h_d(n) = \sum_{i=1}^{P} A_i e^{-\alpha_i n} \cos(\beta_i n + \theta_i) R_N(n) \tag{2.24}$$

The parameters of the time sequence are given by:

$$p = 6, \quad A_i = 1; \quad i = 1,2,...,6, \quad \alpha_1 = \theta_1 = 0,$$

$$\alpha_i = \alpha_{i-1} + \Delta\alpha, \quad \theta_i = \theta_{i-1} + \Delta\theta; \quad i = 2,3,...,6.$$

$$\Delta\alpha = 5/N, \quad \Delta\theta = \frac{\pi}{24},$$

$$\beta_i = (2\pi/N)\cdot m_i, \text{ where}$$

$$m_i = 12, 80, 25, 41, 64, 8 \quad \text{for } i = 1, 2, 3, ..., 6, \quad \text{respectively.}$$

The window size is chosen as 256, i.e., 256 values of $h(n)$ and $h_d(n)$ are chosen to compute equation (2.21). Similarly, three comparisons are made.

a) Let the order of the numerator polynomial in equation (2.1) $M = 11$, the order of the denominator polynomial in equation (2.1) $N = 12$. In this case, the matrix H_2 is full rank.
b) Let $M = 12$, $N = 13$, in this case, the matrix H_2 is rank-deficient.
c) Let $M = 13$, $N = 14$, in this case, the matrix H_2 is rank-deficient.

By using the *Proposed algorithm 4*, the proposed filters are designed.

When the proposed filter order $M = 11$, $N = 12$, the proposed filter coefficients are:

b = 5.5406, -2.9227×10^{1}, 7.6318×10^{1}, -1.3428×10^{2}, 1.8127×10^{2}, -1.9935×10^{2}, 1.8227×10^{2}, -1.3832×10^{2}, 8.5438×10^{1}, -4.0869×10^{1}, 1.3468×10^{1}, -2.2628.

a =1.0000, -5.5240, 1.5270×10^{1}, -2.8868×10^{1}, 4.2666×10^{1}, -5.2391×10^{1}, 5.4569×10^{1}, -4.8284×10^{1}, 3.5994×10^{1}, -2.2067×10^{1}, 1.0498×10^{1}, -3.4125, 5.5658×10^{-1}.

When the proposed filter order $M = 12$, $N = 13$, the proposed filter coefficients are:

b = 5.5406, -2.3938×10^{1}, 4.8416×10^{1}, -6.1419×10^{1}, 5.3088×10^{1}, -2.6300×10^{1}, -8.0359, 3.5687×10^{1}, -4.6608×10^{1}, 4.0694×10^{1}, -2.5547×10^{1}, 1.0595×10^{1}, -2.1602.

a =1.0000, -4.5694, 9.9965, -1.4291×10^{1}, 1.5107×10^{1}, -1.1661×10^{1},

4.5542, 3.8100, -1.0101×10^{1}, 1.2294×10^{1}, -1.0569×10^{1}, 6.6093,

-2.7011, 5.3134×10^{-1}.

When the proposed filter order $M = 13$, $N = 14$, the proposed filter coefficients are:

b = 5.5406, -1.9071×10^{1}, 2.7845×10^{1}, -2.1280×10^{1}, 5.3698, 9.3736,

-1.6350×10^{1}, 1.2370×10^{1}, -3.9906×10^{-1}, -1.1522×10^{1}, 1.7162×10^{1},

-1.5176×10^{1}, 8.2431, -2.0818.

a =1.0000, -3.6911, 6.0650, -5.9614, 3.8017, -7.4707×10^{-1},

-2.2072, 3.5367, -2.3038, -5.1538×10^{-1}, 3.1645, -4.4725, 3.9599, -2.1193,

5.1205×10^{-1}.

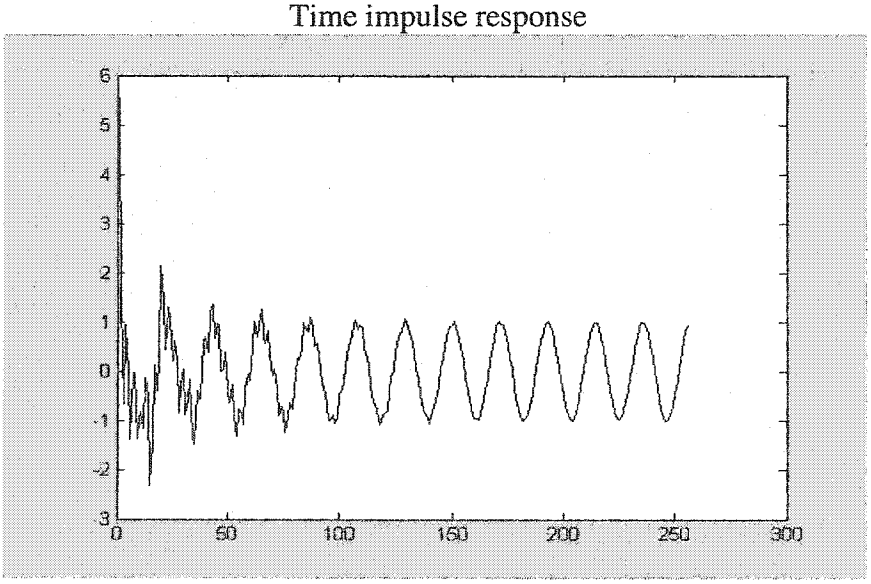


Figure 2.5: Time impulse response of the desired filter (Example 2.3)

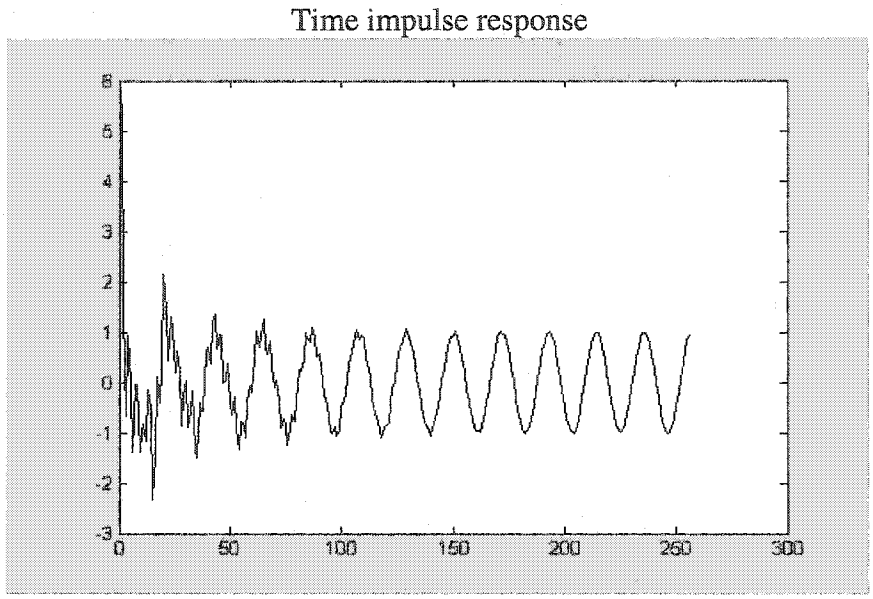


Figure 2.6: Time impulse response of the proposed filter ($M = 11, N = 12$, Example 2.3)

The comparison results $e(n)$ in equation (2.21) are shown in table 2.2.

Table 2.2 The comparison results of example 2.3

	Proposed algorithm 4	Matlab prony( ) function
$e(n)$ $(M = 11, N = 12)$	1.6372×10^{-19}	2.4141×10^{-18}
$e(n)$ $(M = 12, N = 13)$	2.6858×10^{-22}	3.9989×10^{13}
$e(n)$ $(M = 13, N = 14)$	8.8025×10^{-24}	7.5345×10^{12}

When $M = 11$, $N = 12$, H_2 is full rank. In this case, table 2.2 shows that the errors $e(n)$ of the proposed algorithm and *Matlab prony*() function are almost the same. When

$M=12, N=13,$ or $M=13, N=14,$ H_2 is rank-deficient. In these cases, table 2.2 shows that the error $e(n)$ of the proposed algorithm is very small, while the error $e(n)$ of the *Matlab prony*() function is very large. In these cases, *Matlab prony*() function does not work at all, it is divergent.

- Example 2.4

The truncated desired time impulse response of the IIR system is given by:

$$h_d(n)=(0.8\cos\frac{n\pi}{3}+0.9\cos\frac{n\pi}{4}+0.7^n)R_N(n) \tag{2.25}$$

The window size is chosen as 400. Three comparisons are made.

a) Let the order of the numerator polynomial in equation (2.1) $M=13,$ the order of the denominator polynomial in equation (2.1) $N=14$. In this case, the matrix H_2 is rank-deficient.
b) Let $M=14, N=15,$ in this case, the matrix H_2 is rank-deficient.
c) Let $M=15, N=16,$ in this case, the matrix H_2 is rank-deficient.

By using the *Proposed algorithm 4*, the proposed filters are designed.
When the proposed filter order $M=13, N=14$, the proposed filter coefficients are:

$b=2.7000, 9.7247\times10^{-1}, 1.5130\times10^{-1}, 4.3519\times10^{-1}, 8.7475\times10^{-1}, 5.9576\times10^{-1},$
$-2.3784\times10^{-1}, -7.5694\times10^{-1}, -4.9799\times10^{-1}, 7.0398\times10^{-2}, 1.9106\times10^{-1},$
$-2.1375\times10^{-1}, -4.8715\times10^{-1}, -2.1107\times10^{-1}.$

$a=1.0000, -2.8294\times10^{-1}, 2.0466\times10^{-1}, 4.4395\times10^{-1}, 3.0963\times10^{-1}, 3.8482\times10^{-3},$
$-1.8779\times10^{-1}, -1.7306\times10^{-1}, -9.1410\times10^{-2}, -1.0768\times10^{-1}, -2.0733\times10^{-1},$
$-2.2262\times10^{-1}, -5.8277\times10^{-2}, 1.4063\times10^{-1}, 8.5629\times10^{-2}.$

When the proposed filter order $M = 14$, $N = 15$, the proposed filter coefficients are:

$b = 2.7000, 9.2022\times10^{-1}, -6.1691\times10^{-3}, 2.7244\times10^{-1}, 8.3422\times10^{-1}, 6.8426\times10^{-1},$
$-1.2844\times10^{-1}, -7.0349\times10^{-1}, -4.4663\times10^{-1}, 2.2298\times10^{-1}, 4.2861\times10^{-1},$
$-5.7778\times10^{-2}, -5.3423\times10^{-1}, -3.0547\times10^{-1}, 2.7295\times10^{-1}.$

$a = 1.0000, -3.0229\times10^{-1}, 1.5879\times10^{-1}, 4.1383\times10^{-1}, 3.0768\times10^{-1}, 1.2705\times10^{-2},$
$-1.8360\times10^{-1}, -1.6345\times10^{-1}, -5.3036\times10^{-2}, -3.7794\times10^{-2}, -1.3807\times10^{-1},$
$-1.9305\times10^{-1}, -6.7372\times10^{-2}, 1.5162\times10^{-1}, 1.9764\times10^{-1}, -1.1073\times10^{-1}.$

When the proposed filter order $M = 15$, $N = 16$, the proposed filter coefficients are:

$b = 2.7000, 1.0435, -5.7091\times10^{-2}, -7.9372\times10^{-3}, 5.5718\times10^{-1}, 6.5025\times10^{-1},$
$6.0211\times10^{-2}, -5.1476\times10^{-1}, -3.8368\times10^{-1}, 2.9394\times10^{-1}, 7.1019\times10^{-1},$
$3.8432\times10^{-1}, -2.5944\times10^{-1}, -4.1360\times10^{-1}, 9.1095\times10^{-2}, 4.9796\times10^{-1}.$

$a = 1.0000, -2.5663\times10^{-1}, 1.1057\times10^{-1}, 3.3947\times10^{-1}, 3.7299\times10^{-1}, 2.3292\times10^{-2},$
$-1.6727\times10^{-1}, -1.6438\times10^{-1}, -4.3406\times10^{-2}, 2.8041\times10^{-2}, -1.5497\times10^{-2},$
$-7.6165\times10^{-2}, -2.3578\times10^{-2}, 1.3236\times10^{-1}, 2.2409\times10^{-1}, 9.7516\times10^{-2},$
$2.0201\times10^{-1}.$

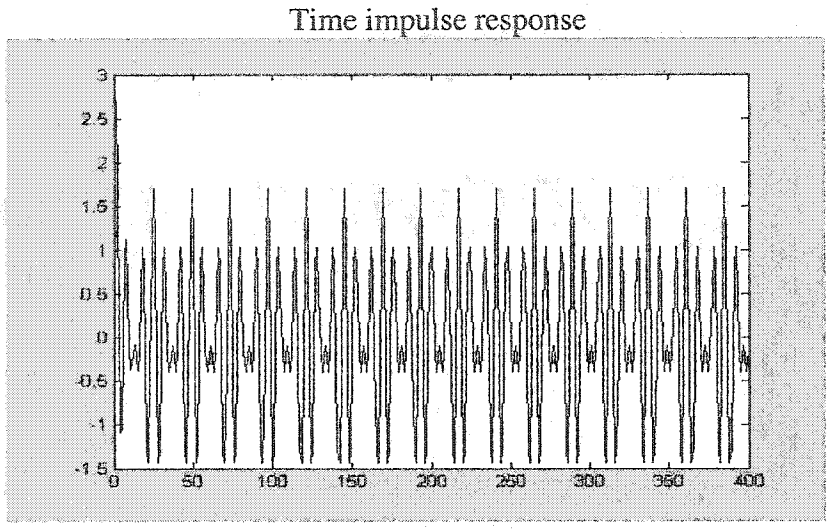


Figure 2.7: Time impulse response of the desired filter (Example 2.4)

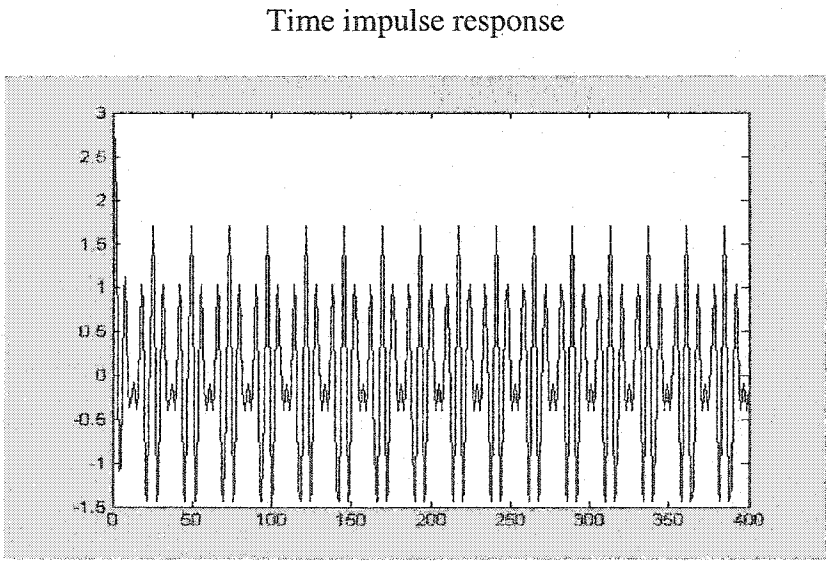


Figure 2.8: Time impulse response of the proposed filter ($M = 13, N = 14$, Example 2.4)

The comparison results are shown in table 2.3.

Table 2.3 The comparison results of example 2.4

	Proposed algorithm 4	Matlab prony( ) function
$e(n)$ $(M=13, N=14)$	4.7684×10^{-26}	2.7601×10^{27}
$e(n)$ $(M=14, N=15)$	4.6264×10^{-26}	7.5238×10^{31}
$e(n)$ $(M=15, N=16)$	4.6148×10^{-26}	1.0867×10^{17}

Table 2.3 shows that the errors $e(n)$ of the proposed algorithm are very small, while the errors $e(n)$ of *Matlab prony*() function are very large. In these cases, *Matlab prony*() function is divergent.

In this chapter, an algorithm ($QR-SVD-PRONY$ algorithm) is proposed in the time domain to design IIR filter. Several design examples are tested to compare the proposed algorithm with *Matlab prony*() function. The test results show that the proposed algorithm has good performance and good convergence, while *Matlab prony*() function in some cases is divergent. Moreover, the proposed algorithm requires no a priori knowledge of the filter order.

CONCLUSION

In this thesis, four algorithms are proposed to design stable IIR filters. Three of them (*Proposed algorithm 1, Proposed algorithm 2, Proposed algorithm 3*) are proposed in the frequency domain based on damped Gauss-Newton method. *Proposed algorithm 3* combines damped Gauss-Newton method and quadratic programming. The fourth algorithm (*Proposed algorithm 4*) is proposed in the time domain based on some numerical analysis techniques (QR factorization and SVD decomposition) and Prony method.

Proposed algorithm 1 approximates the arbitrary magnitude response of the desired filter. *Proposed algorithm 2* and *Proposed algorithm 3* approximate simultaneously the arbitrary magnitude response and phase response of the desired filters with specified stability region. *Proposed algorithm 4* approximates the arbitrary time impulse response of the desired filters.

Several design examples are tested to compare the proposed algorithms with other algorithms. Comparison results show that the proposed algorithms have smaller errors than other algorithms. Moreover, the *Proposed algorithm 1, Proposed algorithm 2, Proposed algorithm 4* have good convergence. The *Proposed algorithm 3* in some cases is not convergent. All these proposed algorithms require no a prior knowledge of the filters order.

REFERENCES

Michael J.Corinthios, "A Fast Z Transformation Algorithm for System Identification." IEEE Trans. Computers, vol. 26, no 1, pp. 55-67. Jan. 1977.

Michael J.Corinthios, "3D Cellular Arrays for Parallel/Cascaded Image/Signal Processing." Chapter 5, "Spectral Analysis and Fault Detection." Mark Karpovsky. Orlando, Fla.: Academic Press. pp.217-298. 1985.

Michael J.Corinthios, "Spectral Analysis on the Complex Plane-Recent Results." Proc. Second Int'l Workshop Spectral Techniques, Montréal, Oct.5-7, 1986.

Michael J.Corinthios, "System Identification by Two Dimensional Spectral Decomposition." Adv. In Comm. And Control Systems, pp. 1171-1178, Baton Rouge, La., Oct. 19-21, 1988.

Michael J.Corinthios, "Pipe-Line Constant Geometry Algorithms and Processors for the Generalized Spectral Analysis of Images." Proc. IMACS-IFAC Symp. Parallel and Distributed Computing, Corfu, Greece, pp. 23-28. June 1991.

Michael J.Corinthios, "Optimal Parallel and Pipelined Processing Through a New Class of Matrices with Application to Generalized Spectral Analysis." IEEE Trans. Computers, vol. 43, no. 4, pp. 443-459. Apr 1994.

Michael J. Corinthios, "A Weighted Z Spectrum, Parallel Algorithm, and Processors for Mathematical Model Estimation." IEEE Trans. Computers, vol. 45, no. 5, pp. 1-16. May 1996.

Natic Mass, Matlab: User's Guide. MathWorks, Inc. 1994.

ANDREAS ANTONIOU, Digital Filters. Analysis, design, and applications. Second edition New York: McGraw-Hill, Inc. 1993.

T.W.Parks, C.S. Burrus, Digital Filter Design. New York: JOHN WILEY & SONS, Inc. 1987.

Leland B. Jackson, Digital Filters and Signal Processing. MA: Kluwer Academic Publishers, 1989.

Alan V. Oppenheim, Ronald W. Schafer, Discrete-Time Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1989.

Mathias C. Lang, "Least-Squares Design of IIR Filters with Prescribed Magnitude and Phase Responses and a Pole Radius." IIEEE Trans. Signal Processing, vol. 48, no. 11, pp. 3109-3121. Nov 2000.

M.C. Lang, "Least squares design of IIR filters with arbitrary magnitude and phase responses and specified stability margin." in Proc. Signal Processing IX (EUSIPCO' 98), vol. III, Rhodes, Greece, Sept. 1998, pp. 1909-1912.

Wu–Sheng Lu, Soo-Chang Pei, Chien-Cheng Tseng, "A weighted least-squares method for the design of stable 1-D and 2-D IIR digital filters." IEEE Trans. Signal Processing, vol. 46, pp. 1-10, Jan. 1998.

Wu–Sheng Lu, "Design of stable IIR digital filters with equiripple passbands and peak-constrained least squares stopbands." In Proc. IEEE Int. Symp. Circuits Syst., vol. 4, Hong Kong, June 1997, pp. 2192-2195.

Y. C. Lim, J. H. Lee, C. K. Chen, and R.H. Yang, "A weighted least squares algorithm for quasi-equiripple FIR and IIR digital filter design." IEEE Trans. Acoust., Speech, Signal Processing, vol. 40, pp. 551-558, Mar. 1992.

T. Kobayashi and S. Imai, "Design of IIR digital filters with arbitrary log magnitude function by WLS technique." IEEE Trans. Acoust., Speech, Signal Processing, vol. 38, pp. 247-252, Feb. 1990.

X. Chen and T. W. Parks, "Design of IIR filters in the complex domain." IEEE Trans. Acoust., Speech, Signal Processing, vol. 38, pp. 910-920, June 1990.

A. G. Deczky, "Synthesis of recursive digital filters using the minimum p-error criterion." IEEE Trans. Audio Electroacoust., vol. AU-20, pp. 257-263, 1972.

S. Alliney and F. Sgallari, "Chebyshev approximation of recursive digital filters having specified amplitude and phase characteristics." Signal Processing, vol. 2. No. 4. pp.317-321. Oct. 1980.

Y. Ishizaki and H. Watanabe, "An iterative Chebyshev approximation method for network design." IEEE Trans. Circuit Theory. vol. CT-15, pp. 326-336, 1968.

S. Ellacott and J. Williams, "Rational Chebyshev approximation in the complex plane." SIAM J. Numer. Anal., vol. 13. No. 3, pp. 310-323, 1976.

J. Williams, "Characterization and computation of rational Chebyshev approximation in the complex plane." SIAM J. Numer. Anal., vol. 16. No. 5, pp. 819-827. 1979.

CHARLES S. BURRUS, THOMAS W. PARKS, "Time Domain Design of Recursive Digital Filters. " IEEE Trans. Audio and Electroacoustics. Vol. AU-18, No. 2 pp.137-141. JUNE 1970.

L.E. McBride, H.W. SCHAEFGEN, K.STEIGLITZ. "Time-Domain Approximation by Iterative Methods." IEEE Trans. Circuit Theory. vol. CT-13. No. 4, pp. 381-387. Dec. 1966.

MONSON H. HAYES, STATISTICAL DIGITAL SIGNAL PROCESSING AND MODELING, New York: John Wiley & Sons, Inc., 1996.

Ashraf Alkhairy, "an Efficient Method for IIR Filter Design." IEEE, 1994.

Tatsuya MATSUNAGA, Masahiro YOSHIDA and Masaaki IKEHARA, "Design of IIR Digital Filters in the Complex Domain by Transforming the Desired Response." IEEE, 2000.

Ne-Zheng Sun, Inverse Problems in Groundwater Modeling. MA: KLUWER ACADEMIC PUBLISHERS, 1994.

R. Fletcher, Practical Methods of Optimization, 2nd ed. Chichester, U. K.: Wiley, 1987.

APPANNA T. CHOTTERA, GREHAM A. JULLIEN, "A Linear Programming Approach to Recursive Digital Filter Design with Linear Phase." IEEE Trans. Circuits and Systems., vol. CAS-29, pp. 139-149, March 1982.

Andrzej Tarczynski, Gerald D. Cain, Ewa Hermanowicz, Miroslaw Rojewski, "STABLE IIR FILTERS – A NEW DESIGN APPROACH." In proc. IEEE Circuits and Systems, ISCAS' 99, vol. 3, pp. 271-274.